<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Related Pages

Here is a list of all related documentation pages:

- Error Codes
- Version History
- Using Skype4COM with Internet Explorer
- Deprecated List

Copyright © 2006 Skype Limited. All rights reserved.
<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Error Codes

- 0x80040200 E_SKYPE_NOT_CONNECTED
- 0x80040201 E_SKYPE_WAIT_TIMEOUT
- 0x80040202 E_SKYPE_WAIT_FAILED
- 0x80040203 E_SKYPE_PROTOCOL_ERROR
- 0x80040204 E_SKYPE_CONNECTION_REFUSED
- 0x80040205 E_SKYPE_PROTOCOL_VERSION
- 0x80040206 E_SKYPE_RUN_FAILED
- 0x80040207 E_SKYPE_NOT_INSTALLED

Copyright © 2006 Skype Limited. All rights reserved.
## Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Version History

1.0.36
- fixed: `IClient::IsRunning` property (supports Skype 5.1 and later builds).

1.0.35
- added UIPI message filter (Windows 7, Vista).

1.0.34
- fixed bug in collection enumerators.

1.0.33
- fixed memory leak in `Skype` class constructor.

1.0.32
- fixed: `IClient::Focus` and open dialog commands did not focus Skype 4.0 window

1.0.30
- fixed: `IClient::IsRunning` property.

1.0.29
- added: `_ISkypeEvents::SilentModeStatusChanged` event.
- added: `_ISkypeEvents::UserAuthorizationRequestReceived` event.
- added: `ISkype::PredictiveDialerCountry` property.
- bugfix: `IChatMember::CanSetRoleTo` always returned VARIANT_FALSE:

1.0.28
- added: `ICall::TargetIdentity` property.
- added: `IChat::AcceptAdd` method.
- added: `IChat::ActivityTimestamp` property.
- added: `IChat::AlertString` property.
- added: `IChat::Applicants` property.
- added: `IChat::Blob` property.
- added: `IChat::ClearRecentMessages` method.
- added: `IChat::Description` property.
- added: `IChat::DialogPartner` property.
- added: `IChat::Disband` method.
- added: `IChat::EnterPassword` method.
- added: `IChat::GuideLines` property.
- added: `IChat::Join` method.
- added: `IChat::Kick` method.
- added: `IChat::KickBan` method.
- added: `IChat::MemberObjects` property.
- added: `IChat::MyRole` property.
- added: `IChat::MyStatus` property.
- added: `IChat::Options` property.
- added: `IChat::PasswordHint` property.
- added: `IChat::SetPassword` method.
- added: `IChat::TopicXML` property.
- added: `IChat::TopicXML` property.
- added: `IChat::Type` property.
- added: `IClient::OpenLiveTab` method.
- added: `ISettings::LoadAvatarFromFile` method.
- added: `ISettings::ResetIdleTimer` method.
- added: `ISettings::SaveAvatarToFile` method.
- added: `ISkype::CreateChatUsingBlob` method.
- added: `ISkype::FindChatUsingBlob` method.
- added: `ISkype::FocusedContacts` property.
- added: `IUser::IsVoicemailCapable` property.
- added: `TChatMemberRole` type.
- added: `TChatMyStatus` type.
- added: `TChatOptions` type.
- added: `TChatType` type.
- added: `_ISkypeEvents::CallTransferStatusChanged` event.
- added: `_ISkypeEvents::ChatMembersChanged` event.
- added: `_ISkypeEvents::ChatMemberRoleChanged` event.
- added: interfaces `IChatMember` and `IChatMemberCollection`.
- change: `ISettings::RingToneStatus` property settable.
- change: calls to event handler are now serialised.
- added: _ISkypeEvents::CallVideoStatusChanged event.
- added: _ISkypeEvents::CallVideoSendStatusChanged event.
- added: _ISkypeEvents::CallVideoReceiveStatusChanged event.
- bugfix: #59913 Microphone stream redirection to port and file

1.0.27
- added: ICall::CanTransfer property.
- added: ICall::TransferStatus property.
- added: ICall::TransferActive property.
- added: ICall::TransferredBy property.
- added: ICall::TransferredTo property.
- added: ICall::Transfer method.
- added: ISkype::ActiveFileTransfers property.
- added: ISkype::FileTransfers property.
- added: _ISkypeEvents::FileTransferStatusChanged event.
- added: IUser::RichMoodText property.
- added: IProfile::RichMoodText property.
- fixed: _ISkypeEvents::GroupUsers event did not fire.

1.0.26
- fixed: bugs in translation file.
- fixed: IDL incompatibility with version 1.0.0.22

1.0.25
- fixed: self-registration for LUA user.
- fixed: ICall::InputDevice and ICall::OutputDevice caching problem.
- fixed: ISkype::Protocol wrong value.
- added: ISkype::ApiWrapperVersion property.
- added: ISkype::SilentMode property.
- added: IClient::Wallpaper property.
- added: _ISkypeEvents::WallpaperChanged event.

1.0.24.0
- added PluginEvent class.
- added PluginMenuItem class.
• added _ISkypeEvents::PluginEventClicked event.
• added _ISkypeEvents::PluginMenuItemClicked event.

1.0.23.0
• changed file versioning format.
• added TChatMessageType::cmeEmoted.
• added _ISkypeEvents::CallSeenStatusChanged event.
• changed ICall::Seen property readable.
• added ICall::ForwardedBy property.
• changed IUser::DisplayName property is now writable.
• fixed: bug in IProfile::CallSendToVM implementation.

1.0.0.22
• added ISkype::AsyncSearchUsers method.
• added ICall::InputDevice property.
• added ICall::InputStatus property.
• added ICall::OutputDevice property.
• added ICall::CaptureMicDevice property.
• added IUser::NumberOfAuthBuddies property.
• added _ISkypeEvents::CallInputStatusChanged event.
• added _ISkypeEvents::AsyncSearchUsersFinished event.

1.0.0.21
• added IClient::OpenSmsDialog method.
• added IProfile::ValidatedSmsNumbers property.
• added ISkype::CreateSms property.
• added ISkype::MissedSmss property.
• added ISkype::SendSms method.
• added ISkype::Smss property.
• added SmsChunk class.
• added SmsChunkCollection class.
• added SmsMessage class.
• added SmsMessageCollection class.
• added SmsTarget class.
• added SmsTargetCollection class.
• fixed: bug in IChatMessage::Seen implementation.

1.0.0.14
• added IProfile::IPCountry property.
• fixed: bug in command cache implementation that caused strings to be truncated.

1.0.0.12
• added ISkype::ApiSecurityContextEnabled property.
• added ISkype::EnableApiSecurityContext method.
• added ISkype::MissedVoicemails property.
• added ICall::Rate property.
• added ICall::RateCurrency property.
• added ICall::RatePrecision property.
• added IChat::Bookmarked property.
• added IChat::Bookmark method.
• added IChat::Unbookmark method.
• added IChatMessage::Sender property.
• added IGroup::Accept method.
• added IGroup::Decline method.
• added IGroup::Share method.
• changed: ISettings::Language writable.
• added IVoicemail::SetUnplayed property.

1.0.0.11
• added CallChannel class.
• added CallChannelCollection class.
• added CallChannelManager class.
• added CallChannelMessage class.

1.0.0.10
• added _ISkypeEvents::UserMood event.

1.0.0.9
• added HTML documentation

Copyright © 2006 Skype Limited. All rights reserved.
<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Using Skype4COM with Internet Explorer

Adding Skype4COM to a web page

To add Skype4COM controls to a web page, use the `OBJECT` tag in the `HTML` code.

```html
<html>
<body>
<object id=Skype
     name=Skype
     height=0
     width=0
     classid=clsid:830690FC-BF2F-47A6-AC2D-330BCB402664
     codebase="http://www.yoursite.com/Skype4COM.dll">
  <span style="color: red">Failed to load control.</span>
</object>
</body>
</html>
```

Installing Skype4COM with Internet Explorer

Before you begin, ensure that Internet Explorer security policies are set to download signed ActiveX controls. Select Tools menu > Internet Options > Security tab and click the Custom Level button to display the ActiveX settings.
If downloading ActiveX controls is set to Prompt, when a user first visits a web page containing a Skype4COM object, Internet Explorer displays a warning.

The user must click the Install button to start the download. After the installation, the Internet Explorer page reloads, and Skype4COM is ready for use.

**Skype4COM security policies for Internet Explorer**
Internet Explorer defines five security zones for web content:

- Local Machine (this zone is not visible in the Internet Explorer security tab)
- Local Intranet
- Trusted
- Internet
- Restricted

Skype4COM defines four API security contexts:

- Voice services
- Messaging services
- Account services
- Contact services

During installation, Skype4COM creates custom policies for each API context in each Internet Explorer security zone.

<table>
<thead>
<tr>
<th>Security Zone</th>
<th>Voice API</th>
<th>Messaging API</th>
<th>Account API</th>
<th>Contact API</th>
</tr>
</thead>
<tbody>
<tr>
<td>Local Machine</td>
<td>Allow</td>
<td>Allow</td>
<td>Allow</td>
<td>Allow</td>
</tr>
<tr>
<td>Local Intranet</td>
<td>Allow</td>
<td>Allow</td>
<td>Prompt</td>
<td>Allow</td>
</tr>
<tr>
<td>Trusted</td>
<td>Allow</td>
<td>Allow</td>
<td>Prompt</td>
<td>Allow</td>
</tr>
<tr>
<td>Internet</td>
<td>Prompt</td>
<td>Prompt</td>
<td>Prompt</td>
<td>Prompt</td>
</tr>
<tr>
<td>Restricted</td>
<td>Disallow</td>
<td>Disallow</td>
<td>Disallow</td>
<td>Disallow</td>
</tr>
</tbody>
</table>

If the zone setting for a security context is set to "Prompt", Skype4COM displays a window where the user must select whether to allow this context or not. For example, if a web page
contains a Call Me button, and voice services for the Internet security zone are set to prompt, the following window appears:

![Window for Skype voice services](image)

The user receives multiple prompts if a web page contains more than one Skype API security context, for example a page with a Call Me button and a Chat to Me button, where the security policies for voice and messaging services are both set to Prompt.

If the user selects the Allow (first) or Disallow (last) option, the preference is saved as a host setting. When the user next visits this web page, Internet Explorer compares the security policies for the zone and the host, and applies the most secure policy. For example, if security for the zone is set to Prompt, and security for the host is set to Disable (user selected Disable on a previous visit to the web page), Internet Explorer applies the most secure policy, does not display a prompt window to the user, and does not enable the service.

**Managing security policies**

Custom security policies for web content zones are stored in the Windows registry for the current user. Skype4COM provides a window where you can view and change these settings. Type `skype4com:openzonesettings` into the Internet Explorer address bar to access the zone security settings.
Custom security policies entered by the user in prompt windows are stored in Windows registry for the current user. Skype4COM provides a window where you can view and change these settings. Type `skype4com:openhostsettings` into the Internet Explorer address bar to access the host security settings.

Copyright © 2006 Skype Limited. All rights reserved.
Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Deprecated List

**Class Conversion**
The SKYPE4COMLib::Conversion class is deprecated, support for multilanguage status codes support will be dropped in the future.

**Member ICall::Status**
Set ICall::Status is deprecated, use the ICall::Hold, ICall::Resume, ICall::Finish or ICall::Answer instead.

**Member IChat::Status**
ICall::Status is deprecated, use the IChat::Type and IChat::MyStatus instead.

**Member ISettings::Avatar ([in, defaultvalue("1")]) BSTR Id**
Use the ISettings::LoadAvatarFromFile instead.

Copyright � 2006 Skype Limited. All rights reserved.
Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Modules

Here is a list of all modules:

- Skype4COM

Copyright 2006 Skype Limited. All rights reserved.
### Typedefs

<table>
<thead>
<tr>
<th>Typedef</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>typedef enum</td>
<td><code>TAttachmentStatus</code></td>
</tr>
<tr>
<td></td>
<td>TAttachmentStatus</td>
</tr>
<tr>
<td>typedef enum</td>
<td><code>TConnectionStatus</code></td>
</tr>
<tr>
<td></td>
<td>TConnectionStatus</td>
</tr>
<tr>
<td>typedef enum</td>
<td><code>TUserStatus</code></td>
</tr>
<tr>
<td></td>
<td>TUserStatus</td>
</tr>
<tr>
<td>typedef enum</td>
<td><code>TCallFailureReason</code></td>
</tr>
<tr>
<td></td>
<td>TCallFailureReason</td>
</tr>
<tr>
<td>typedef enum</td>
<td><code>TCallStatus</code></td>
</tr>
<tr>
<td></td>
<td>TCallStatus</td>
</tr>
<tr>
<td>typedef enum</td>
<td><code>TCallType</code></td>
</tr>
<tr>
<td></td>
<td>TCallType</td>
</tr>
<tr>
<td>typedef enum</td>
<td><code>TCallHistory</code></td>
</tr>
<tr>
<td></td>
<td>TCallHistory</td>
</tr>
<tr>
<td>typedef enum</td>
<td><code>TCallVideoStatus</code></td>
</tr>
<tr>
<td></td>
<td>TCallVideoStatus</td>
</tr>
<tr>
<td>typedef enum</td>
<td><code>TCallVideoSendStatus</code></td>
</tr>
<tr>
<td></td>
<td>TCallVideoSendStatus</td>
</tr>
<tr>
<td>typedef enum</td>
<td><code>TCallIoDeviceType</code></td>
</tr>
<tr>
<td></td>
<td>TCallIoDeviceType</td>
</tr>
<tr>
<td>typedef enum</td>
<td><code>TChatMessageType</code></td>
</tr>
<tr>
<td></td>
<td>TChatMessageType</td>
</tr>
<tr>
<td>typedef enum</td>
<td><code>TChatMessageStatus</code></td>
</tr>
<tr>
<td></td>
<td>TChatMessageStatus</td>
</tr>
<tr>
<td>typedef enum</td>
<td><code>TChatMemberRole</code></td>
</tr>
<tr>
<td></td>
<td>TChatMemberRole</td>
</tr>
<tr>
<td>typedef enum</td>
<td><code>TUserSex</code></td>
</tr>
<tr>
<td></td>
<td>TUserSex</td>
</tr>
<tr>
<td>typedef enum</td>
<td><code>TBuddyStatus</code></td>
</tr>
<tr>
<td></td>
<td>TBuddyStatus</td>
</tr>
<tr>
<td>typedef enum</td>
<td><code>TOnlineStatus</code></td>
</tr>
<tr>
<td></td>
<td>TOnlineStatus</td>
</tr>
<tr>
<td>typedef enum</td>
<td><code>TChatLeaveReason</code></td>
</tr>
<tr>
<td></td>
<td>TChatLeaveReason</td>
</tr>
<tr>
<td>typedef enum</td>
<td><code>TChatStatus</code></td>
</tr>
<tr>
<td></td>
<td>TChatStatus</td>
</tr>
<tr>
<td>typedef enum</td>
<td><code>TChatType</code></td>
</tr>
<tr>
<td></td>
<td>TChatType</td>
</tr>
<tr>
<td>typedef enum</td>
<td><code>TChatMyStatus</code></td>
</tr>
<tr>
<td></td>
<td>TChatMyStatus</td>
</tr>
<tr>
<td>typedef enum</td>
<td><code>TChatOptions</code></td>
</tr>
<tr>
<td></td>
<td>TChatOptions</td>
</tr>
<tr>
<td>typedef enum</td>
<td><code>TVoicemailType</code></td>
</tr>
<tr>
<td></td>
<td>TVoicemailType</td>
</tr>
<tr>
<td>typedef enum</td>
<td><code>TVoicemailStatus</code></td>
</tr>
<tr>
<td></td>
<td>TVoicemailStatus</td>
</tr>
<tr>
<td>typedef enum</td>
<td><code>TVoicemailFailureReason</code></td>
</tr>
<tr>
<td></td>
<td>TVoicemailFailureReason</td>
</tr>
<tr>
<td>typedef enum</td>
<td><code>TGroupType</code></td>
</tr>
<tr>
<td></td>
<td>TGroupType</td>
</tr>
<tr>
<td>typedef enum</td>
<td><code>TCallChannelType</code></td>
</tr>
<tr>
<td></td>
<td>TCallChannelType</td>
</tr>
<tr>
<td>typedef enum</td>
<td><code>TApiSecurityContext</code></td>
</tr>
<tr>
<td></td>
<td>TApiSecurityContext</td>
</tr>
<tr>
<td>typedef enum</td>
<td><code>TSmsMessageType</code></td>
</tr>
<tr>
<td></td>
<td>TSmsMessageType</td>
</tr>
<tr>
<td>typedef enum</td>
<td><code>TSmsMessageStatus</code></td>
</tr>
<tr>
<td></td>
<td>TSmsMessageStatus</td>
</tr>
<tr>
<td>typedef enum</td>
<td><code>TSmsFailureReason</code></td>
</tr>
<tr>
<td></td>
<td>TSmsFailureReason</td>
</tr>
<tr>
<td>typedef enum</td>
<td><code>TSmsTargetStatus</code></td>
</tr>
<tr>
<td></td>
<td>TSmsTargetStatus</td>
</tr>
<tr>
<td>typedef enum</td>
<td><code>TPluginContext</code></td>
</tr>
<tr>
<td></td>
<td>TPluginContext</td>
</tr>
<tr>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>----------------------------</td>
</tr>
<tr>
<td><code>typedef enum TPluginContactType</code></td>
<td><code>TPluginContactType</code></td>
</tr>
<tr>
<td><code>typedef enum TFileTransferType</code></td>
<td><code>TFileTransferType</code></td>
</tr>
<tr>
<td><code>typedef enum TFileTransferStatus</code></td>
<td><code>TFileTransferStatus</code></td>
</tr>
<tr>
<td><code>typedef enum TFileTransferFailureReason</code></td>
<td><code>TFileTransferFailureReason</code></td>
</tr>
</tbody>
</table>
## Enumerations

<table>
<thead>
<tr>
<th>Enum</th>
<th>TAttachmentStatus</th>
<th>TConnectionStatus</th>
<th>TUserStatus</th>
<th>TCallFailureReason</th>
<th>TCallStatus</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>apiAttachUnknown = -1, apiAttachSuccess = 0, apiAttachPendingAuthorization = 1, apiAttachRefused = 2, apiAttachNotAvailable = 3, apiAttachAvailable = 4</td>
<td>conUnknown = -1, conOffline = 0, conConnecting = 1, conPausing = 2, conOnline = 3</td>
<td>cusUnknown = -1, cusOffline = 0, cusOnline = 1, cusAway = 2, cusNotAvailable = 3, cusDoNotDisturb = 4, cusInvisible = 5, cusLoggedOut = 6, cusSkypeMe = 7</td>
<td>cfrUnknown = -1, cfrMiscError = 0, cfrUserDoesNotExist = 1, cfrUserIsOffline = 2, cfrNoProxyFound = 3, cfrSessionTerminated = 4, cfrNoCommonCodec = 5, cfrSoundIODeviceError = 6, cfrRemoteDeviceError = 7, cfrBlockedByRecipient = 8, cfrRecipientNotFriend = 9, cfrNotAuthorizedByRecipient = 10, cfrSoundRecordingError = 11</td>
<td>clsUnknown = -1, clsUnplaced = 0, clsRouting = 1, clsEarlyMedia = 2, clsFailed = 3, clsRinging = 4, clsInProgress = 5, clsOnHold = 6, clsFinished = 7, clsMissed = 8, clsRefused = 9, clsBusy = 10</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Enum</td>
<td>Values</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---------------------------</td>
<td>--------------------------------------------</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>clsCancelled = 11</td>
<td>clsLocalHold = 12, clsRemoteHold = 13</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>clsVoicemailBufferingGreeting = 14,</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>clsVoicemailPlayingGreeting = 15,</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>clsVoicemailRecording = 16, clsVoicemailUploading = 17,</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>clsVoicemailSent = 18, clsVoicemailCancelled = 19,</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>clsVoicemailFailed = 20, clsTransferring = 21,</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>clsTransferred = 22</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TCallType</td>
<td>cltUnknown = -1, cltIncomingPSTN = 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>cltOutgoingPSTN = 1, cltIncomingP2P = 2,</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>cltOutgoingP2P = 3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TCallHistory</td>
<td>chsAllCalls = 0, chsMissedCalls = 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>chsIncomingCalls = 2, chsOutgoingCalls = 3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TCallVideoStatus</td>
<td>cvsUnknown = -1, cvsNone = 0, csvSendEnabled = 1,</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>csvReceiveEnabled = 2, csvBothEnabled = 3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TCallVideoSendStatus</td>
<td>vssUnknown = -1, vssNotAvailable = 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>vssAvailable = 1, vssStarting = 2,</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>vssRejected = 3, vssRunning = 4, vssStopping = 5,</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>vssPaused = 6</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TCallIoDeviceType</td>
<td>callIoDeviceTypeUnknown = -1, callIoDeviceTypeSoundcard = 0,</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>callIoDeviceTypePort = 1, callIoDeviceTypeFile = 2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TChatMessageType</td>
<td>cmeUnknown = -1, cmeCreatedChatWith = 0,</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>cmeSawMembers = 1, cmeAddedMembers = 2,</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>cmeSetTopic = 3, cmeSaid = 4, cmeLeft = 5,</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>cmeEmoted = 6, cmePostedContacts = 7</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>cmeGapInChat = 8, cmeSetRole = 9, cmeKicked = 10,</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>cmeSetOptions = 11, cmeKickBanned = 12,</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>cmeJoinedAsApplicant = 13, cmeSetPicture = 14,</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```plaintext
cmeSetGuidelines = 15

<table>
<thead>
<tr>
<th>Enum</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>TChatMessageStatus</td>
<td>cmsUnknown = -1, cmsSending = 0, cmsSent = 1, cmsReceived = 2, cmsRead = 3</td>
</tr>
<tr>
<td>TChatMemberRole</td>
<td>chatMemberRoleUnknown = -1, chatMemberRoleCreator = 0, chatMemberRoleMaster = 1, chatMemberRoleHelper = 2, chatMemberRoleUser = 3, chatMemberRoleListener = 4, chatMemberRoleApplicant = 5</td>
</tr>
<tr>
<td>TUUserSex</td>
<td>usexUnknown = -1, usexMale = 0, usexFemale = 1</td>
</tr>
<tr>
<td>TBuddyStatus</td>
<td>budUnknown = -1, budNeverBeenFriend = 0, budDeletedFriend = 1, budPendingAuthorization = 2, budFriend = 3</td>
</tr>
<tr>
<td>TOnlineStatus</td>
<td>olsUnknown = -1, olsOffline = 0, olsOnline = 1, olsAway = 2, olsNotAvailable = 3, olsDoNotDisturb = 4, olsSkypeOut = 5, olsSkypeMe = 6</td>
</tr>
<tr>
<td>TChatLeaveReason</td>
<td>leaUnknown = -1, leaUserNotFound = 0, leaUserIncapable = 1, leaAdderNotFriend = 2, leaAddedNotAuthorized = 3, leaAddDeclined = 4, leaUnsubscribe = 5</td>
</tr>
<tr>
<td>TChatStatus</td>
<td>chsUnknown = -1, chsLegacyDialog = 0, chsDialog = 1, chsMultiNeedAccept = 2, chsMultiSubscribed = 3, chsUnsubscribed = 4</td>
</tr>
</tbody>
</table>
```
<table>
<thead>
<tr>
<th>Enum</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>TChatType</code></td>
<td>Enum for chat types:</td>
</tr>
<tr>
<td></td>
<td>- <code>chatTypeUnknown</code> = -1</td>
</tr>
<tr>
<td></td>
<td>- <code>chatTypeDialog</code> = 0</td>
</tr>
<tr>
<td></td>
<td>- <code>chatTypeLegacyDialog</code> = 1</td>
</tr>
<tr>
<td></td>
<td>- <code>chatTypeLegacyUnsubscribed</code> = 2</td>
</tr>
<tr>
<td></td>
<td>- <code>chatTypeMultiChat</code> = 3</td>
</tr>
<tr>
<td></td>
<td>- <code>chatTypeSharedGroup</code> = 4</td>
</tr>
<tr>
<td><code>TChatMyStatus</code></td>
<td>Enum for chat status:</td>
</tr>
<tr>
<td></td>
<td>- <code>chatStatusUnknown</code> = -1</td>
</tr>
<tr>
<td></td>
<td>- <code>chatStatusConnecting</code> = 0</td>
</tr>
<tr>
<td></td>
<td>- <code>chatStatusWaitingRemoteAccept</code> = 1</td>
</tr>
<tr>
<td></td>
<td>- <code>chatStatusAcceptRequired</code> = 2</td>
</tr>
<tr>
<td></td>
<td>- <code>chatStatusPasswordRequired</code> = 3</td>
</tr>
<tr>
<td></td>
<td>- <code>chatStatusSubscribed</code> = 4</td>
</tr>
<tr>
<td></td>
<td>- <code>chatStatusUnsubscribed</code> = 5</td>
</tr>
<tr>
<td></td>
<td>- <code>chatStatusDisbanded</code> = 6</td>
</tr>
<tr>
<td></td>
<td>- <code>chatStatusQueuedBecauseChatIsFull</code> = 7</td>
</tr>
<tr>
<td></td>
<td>- <code>chatStatusApplicationDenied</code> = 8</td>
</tr>
<tr>
<td></td>
<td>- <code>chatStatusKicked</code> = 9</td>
</tr>
<tr>
<td></td>
<td>- <code>chatStatusBanned</code> = 10</td>
</tr>
<tr>
<td></td>
<td>- <code>chatStatusRetryConnecting</code> = 11</td>
</tr>
<tr>
<td><code>TChatOptions</code></td>
<td>Enum for chat options:</td>
</tr>
<tr>
<td></td>
<td>- <code>chatOptionJoiningEnabled</code> = 1</td>
</tr>
<tr>
<td></td>
<td>- <code>chatOptionJoinersBecomeApplicants</code> = 2</td>
</tr>
<tr>
<td></td>
<td>- <code>chatOptionJoinersBecomeListeners</code> = 4</td>
</tr>
<tr>
<td></td>
<td>- <code>chatOptionHistoryDisclosed</code> = 8</td>
</tr>
<tr>
<td></td>
<td>- <code>chatOptionUsersAreListeners</code> = 16</td>
</tr>
<tr>
<td></td>
<td>- <code>chatOptionTopicAndPictureLockedForUsers</code> = 32</td>
</tr>
<tr>
<td><code>TVoicemailType</code></td>
<td>Enum for voicemail types:</td>
</tr>
<tr>
<td></td>
<td>- <code>vmtUnknown</code> = -1</td>
</tr>
<tr>
<td></td>
<td>- <code>vmtIncoming</code> = 0</td>
</tr>
<tr>
<td></td>
<td>- <code>vmtDefaultGreeting</code> = 1</td>
</tr>
<tr>
<td></td>
<td>- <code>vmtCustomGreeting</code> = 2</td>
</tr>
<tr>
<td></td>
<td>- <code>vmtOutgoing</code> = 3</td>
</tr>
<tr>
<td><code>TVoicemailStatus</code></td>
<td>Enum for voicemail status:</td>
</tr>
<tr>
<td></td>
<td>- <code>vmsUnknown</code> = -1</td>
</tr>
<tr>
<td></td>
<td>- <code>vmsNotDownloaded</code> = 0</td>
</tr>
<tr>
<td></td>
<td>- <code>vmsDownloading</code> = 1</td>
</tr>
<tr>
<td></td>
<td>- <code>vmsUnplayed</code> = 2</td>
</tr>
<tr>
<td></td>
<td>- <code>vmsBuffering</code> = 3</td>
</tr>
<tr>
<td></td>
<td>- <code>vmsPlaying</code> = 4</td>
</tr>
<tr>
<td></td>
<td>- <code>vmsPlayed</code> = 5</td>
</tr>
<tr>
<td></td>
<td>- <code>vmsBlank</code> = 6</td>
</tr>
<tr>
<td></td>
<td>- <code>vmsRecording</code> = 7</td>
</tr>
<tr>
<td></td>
<td>- <code>vmsRecorded</code> = 8</td>
</tr>
<tr>
<td></td>
<td>- <code>vmsUploading</code> = 9</td>
</tr>
<tr>
<td></td>
<td>- <code>vmsUploaded</code> = 10</td>
</tr>
</tbody>
</table>
enum TVoicemailFailureReason {
    vmrUnknown = -1, vmrNoError = 0, vmrMiscError = 1,
    vmrConnectError = 2,
    vmrNoPrivilege = 3, vmrNoVoicemail = 4,
    vmrFileReadError = 5, vmrFileWriteError = 6,
    vmrRecordingError = 7, vmrPlaybackError = 8
}

enum TGroupType {
    grpUnknown = -1, grpCustomGroup = 0, grpAllUsers = 1,
    grpAllFriends = 2,
    grpSkypeFriends = 3, grpSkypeOutFriends = 4,
    grpOnlineFriends = 5, grpPendingAuthorizationFriends = 6,
    grpRecentlyContactedUsers = 7, grpUsersWaitingMyAuthorization = 8,
    grpUsersAuthorizedByMe = 9, grpUsersBlockedByMe = 10,
    grpUngroupedFriends = 11, grpSharedGroup = 12,
    grpProposedSharedGroup = 13
}

enum TCallChannelType {
    cctUnknown = -1, cctDatagram = 0, cctReliable = 1
}

enum TApiSecurityContext {
    apiContextUnknown = 0x0000, apiContextVoice = 0x0001,
    apiContextMessaging = 0x0002, apiContextAccount = 0x0004,
    apiContextContacts = 0x0008
}

enum TSmsMessageType {
    smsMessageTypeUnknown = -1, smsMessageTypeIncoming = 0,
    smsMessageTypeOutgoing = 1, smsMessageTypeCCRequest = 2,
    smsMessageTypeCCSubmit = 3
}

TSmsMessageStatus { vmsDeleting = 11, vmsFailed = 12 }
enum smsMessageStatusUnknown = -1,
smsMessageStatusReceived = 0,
smsMessageStatusRead = 1,
smsMessageStatusComposing = 2,
smsMessageStatusSendingToServer = 3,
smsMessageStatusSentToServer = 4,
smsMessageStatusDelivered = 5,
smsMessageStatusSomeTargetsFailed = 6,
smsMessageStatusFailed = 7
}

enum TSmsFailureReason {
    smsFailureReasonUnknown = -1,
    smsFailureReasonMiscError = 0,
    smsFailureReasonServerConnectFailed = 1,
    smsFailureReasonNoSmsCapability = 2,
    smsFailureReasonInsufficientFunds = 3,
    smsFailureReasonInvalidConfirmationCode = 4,
    smsFailureReasonUserBlocked = 5,
    smsFailureReasonIPBlocked = 6,
    smsFailureReasonNodeBlocked = 7
}

enum TSmsTargetStatus {
    smsTargetStatusUnknown = -1,
    smsTargetStatusUndefined = 0,
    smsTargetStatusAnalyzing = 1,
    smsTargetStatusAcceptable = 2,
    smsTargetStatusNotRoutable = 3,
    smsTargetStatusDeliveryPending = 4,
    smsTargetStatusDeliverySuccessful = 5,
    smsTargetStatusDeliveryFailed = 6
}

enum TPluginContext {
    pluginContextUnknown = -1, pluginContextChat = 0,
    pluginContextCall = 1, pluginContextContact = 2,
    pluginContextMyself = 3, pluginContextTools = 4
}

enum TPluginContactType {
    pluginContactTypeUnknown = -1,
    pluginContactTypeAll = 0, pluginContactTypeSkype = 1,
<table>
<thead>
<tr>
<th>pluginContactTypeSkypeOut = 2</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>enum</strong> TFileTransferType { fileTransferTypeIncoming = 0, fileTransferTypeOutgoing = 1 }</td>
</tr>
<tr>
<td><strong>enum</strong> TFileTransferStatus { fileTransferStatusNew = 0, fileTransferStatusConnecting = 1, fileTransferStatusWaitingForAccept = 2, fileTransferStatusTransferring = 3, fileTransferStatusTransferringOverRelay = 4, fileTransferStatusPaused = 5, fileTransferStatusRemotelyPaused = 6, fileTransferStatusCancelled = 7, fileTransferStatusCompleted = 8, fileTransferStatusFailed = 9 }</td>
</tr>
<tr>
<td><strong>enum</strong> TFileTransferFailureReason { fileTransferFailureReasonSenderNotAuthorized = 1, fileTransferFailureReasonRemotelyCancelled = 2, fileTransferFailureReasonFailedRead = 3, fileTransferFailureReasonFailedRemoteRead = 4, fileTransferFailureReasonFailedWrite = 5, fileTransferFailureReasonFailedRemoteWrite = 6, fileTransferFailureReasonRemoteDoesNotSupportFT = 7, fileTransferFailureReasonRemoteOfflineTooLong = 8 }</td>
</tr>
</tbody>
</table>
Detailed Description

Skype4COM is a COM library that enables you to develop applications using the Skype API in a range of ActiveX programming environments and scripting languages.
Enumeration Type Documentation

enum TApiSecurityContext

contains API security contexts for Internet Explorer.

Enumerator:

apiContextUnknown indicates that the API security context is unknown.

apiContextVoice relates to call and voicemail commands and events

apiContextMessaging relates to text and app2app messaging commands and events

apiContextAccount relates to user profile and global settings commands and events

apiContextContacts relates to contacts commands and events

enum TAttachmentStatus

contains the status of an attachment with the Skype API.

Enumerator:

apiAttachUnknown indicates that the attachment status is unknown.
<table>
<thead>
<tr>
<th>API Call</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>apiAttachSuccess</code></td>
<td>indicates that the client is successfully attached.</td>
</tr>
<tr>
<td><code>apiAttachPendingAuthorization</code></td>
<td>indicates that Skype has acknowledged a connection request and is waiting for user authorisation.</td>
</tr>
<tr>
<td><code>apiAttachRefused</code></td>
<td>indicates that a user has explicitly denied access to the client.</td>
</tr>
<tr>
<td><code>apiAttachNotAvailable</code></td>
<td>indicates that the Skype API is not available.</td>
</tr>
<tr>
<td><code>apiAttachAvailable</code></td>
<td>indicates that the Skype API is available.</td>
</tr>
</tbody>
</table>

**enum TBuddyStatus**

contains the status of a user.

**Enumerator:**

- `budUnknown` indicates the status of a contact is unknown.
- `budNeverBeenFriend` indicates the contact has never been in the user's contactlist.


\textit{budDeletedFriend} indicates the contact was deleted from the contact list.

\textit{budPendingAuthorization} indicates the contact is pending authorization.

\textit{budFriend} indicates the contact is in a contact list.

\textbf{enum TCallChannelType} contains the call channel type.

\textbf{Enumerator:}

\textit{cctUnknown} indicates that the call channel type is unknown.

\textit{cctDatagram} indicates that the call channel is a datagram type channel.

\textit{cctReliable} indicates that the call channel is a reliable type channel.

\textbf{enum TCallFailureReason} contains the reason a call fails.

\textbf{Enumerator:}

\textit{cfrUnknown} indicates the reason for call failure is unknown.
<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>cfrMiscError</code></td>
<td>indicates the call failed because of a miscellaneous error.</td>
</tr>
<tr>
<td><code>cfrUserDoesNotExist</code></td>
<td>indicates the call failed because the user or phone number does not exist.</td>
</tr>
<tr>
<td><code>cfrUserIsOffline</code></td>
<td>indicates the call failed because the user is offline.</td>
</tr>
<tr>
<td><code>cfrNoProxyFound</code></td>
<td>indicates the call failed because no proxy was found.</td>
</tr>
<tr>
<td><code>cfrSessionTerminated</code></td>
<td>indicates the call failed because the session terminated.</td>
</tr>
<tr>
<td><code>cfrNoCommonCodec</code></td>
<td>indicates the call failed because no common codec was found.</td>
</tr>
<tr>
<td><code>cfrSoundIOError</code></td>
<td>indicates the call failed because of an audio I/O error.</td>
</tr>
<tr>
<td><code>cfrRemoteDeviceError</code></td>
<td>indicates the call failed because of a problem with the remote sound device.</td>
</tr>
<tr>
<td><code>cfrBlockedByRecipient</code></td>
<td>indicates the recipient blocked the call.</td>
</tr>
</tbody>
</table>
**cfrRecipientNotFriend** indicates the call failed because the recipient is not a friend.

**cfrNotAuthorizedByRecipient** indicates the call failed because the user is not authorized by the recipient.

**cfrSoundRecordingError** indicates the call failed because of a sound recording error.

**enum TCallHistory**

defines call types in call history.

**Enumerator:**

**chsAllCalls** indicates all calls in call history.

**chsMissedCalls** indicates all missed calls in call history.

**chsIncomingCalls** indicates all incoming calls in call history.

**chsOutgoingCalls** indicates all outgoing calls in call history.

**enum TCallIoDeviceType**

contains the call io device types.

**Enumerator:**
```markdown
<table>
<thead>
<tr>
<th>callIoDeviceTypeUnknown</th>
<th>indicates that device is unknown.</th>
</tr>
</thead>
<tbody>
<tr>
<td>callIoDeviceTypeSoundcard</td>
<td>indicates soundcard.</td>
</tr>
<tr>
<td>callIoDeviceTypePort</td>
<td>indicates local port.</td>
</tr>
<tr>
<td>callIoDeviceTypeFile</td>
<td>indicates local file.</td>
</tr>
</tbody>
</table>

**enum TCallStatus**

contains the status of a call.

**Enumerator:**

- **clsUnknown** indicates the call status is unknown.
- **clsUnplaced** indicates the call was not placed.
- **clsRouting** indicates the call is in the process of being routed.
- **clsEarlyMedia** indicates that early media is currently being played, such as recorded speech.
- **clsFailed** indicates the call failed.
```
<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>clsRinging</td>
<td>indicates the call is in the process of ringing.</td>
</tr>
<tr>
<td>clsInProgress</td>
<td>indicates the call is in progress.</td>
</tr>
<tr>
<td>clsOnHold</td>
<td>indicates the call is placed on hold.</td>
</tr>
<tr>
<td>clsFinished</td>
<td>indicates the call is finished.</td>
</tr>
<tr>
<td>clsMissed</td>
<td>indicates the call was missed.</td>
</tr>
<tr>
<td>clsRefused</td>
<td>indicates the call was refused.</td>
</tr>
<tr>
<td>clsBusy</td>
<td>indicates the call destination was busy.</td>
</tr>
<tr>
<td>clsCancelled</td>
<td>indicates the call was cancelled.</td>
</tr>
<tr>
<td>clsLocalHold</td>
<td>indicates the call is placed on hold by the local user.</td>
</tr>
<tr>
<td>clsRemoteHold</td>
<td>indicates the call is placed on hold by the remote user.</td>
</tr>
<tr>
<td>clsVoicemailBufferingGreeting</td>
<td>indicates that voicemail is preparing a greeting.</td>
</tr>
<tr>
<td></td>
<td>indicates that voicemail is</td>
</tr>
<tr>
<td>Class Name</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>-------------</td>
</tr>
<tr>
<td>clsVoicemailPlayingGreeting</td>
<td>playing a greeting.</td>
</tr>
<tr>
<td>clsVoicemailRecording</td>
<td>indicates that voicemail is recording a message.</td>
</tr>
<tr>
<td>clsVoicemailUploading</td>
<td>indicates that voicemail is being loaded to the server.</td>
</tr>
<tr>
<td>clsVoicemailSent</td>
<td>indicates that voicemail has been sent.</td>
</tr>
<tr>
<td>clsVoicemailCancelled</td>
<td>indicates that voicemail has been cancelled.</td>
</tr>
<tr>
<td>clsVoicemailFailed</td>
<td>indicates that voicemail failed.</td>
</tr>
<tr>
<td>clsTransferring</td>
<td>indicates that call is being transferred.</td>
</tr>
<tr>
<td>clsTransferred</td>
<td>indicates that call has been transferred.</td>
</tr>
</tbody>
</table>

**enum TCallType**

contains information about call types.

**Enumerator:**

- **cltUnknown** indicates the call type is unknown.
**cltIncomingPSTN** indicates the call is an incoming call from the PSTN.

**cltOutgoingPSTN** indicates the call is an outgoing call to the PSTN.

**cltIncomingP2P** indicates the call is an incoming Skype call.

**cltOutgoingP2P** indicates the call is an outgoing Skype call.

---

**enum TCallVideoSendStatus**

contains the status for video send streams.

**Enumerator:**

- **vssUnknown** indicates that video send status is unknown.

- **vssNotAvailable** indicates that video send is not available because video is disabled or a webcam is unplugged.

- **vssAvailable** indicates that video send is available but the video is not running.

- **vssStarting** indicates that video is sending but not yet running at full speed.

- **vssRejected** indicates that video send was rejected by the receiver.
vssRunning indicates that video send is actively running.

vssStopping indicates that video send is in the process of stopping but not halted yet.

vssPaused indicates that video send is placed on hold.

enum TCallVideoStatus
contains the video status of a call.

Enumerator:

cvsUnknown indicates that call video status is unknown.

cvsNone indicates that call video status is that video is not enabled.

cvsSendEnabled indicates that call video status is that video send is enabled.

cvsReceiveEnabled indicates that call video status is that video receive is enabled.

cvsBothEnabled indicates that call video status is that video send and receive are enabled.
enum \texttt{TChatLeaveReason} contains the reason a user leaves a chat.

\textbf{Enumerator:}

<table>
<thead>
<tr>
<th>Subscript</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\texttt{leaUnknown }</td>
<td>indicates the reason for leaving a chat is unknown.</td>
</tr>
<tr>
<td>\texttt{leaUserNotFound}</td>
<td>indicates the reason for leaving a chat is that a user was not found.</td>
</tr>
<tr>
<td>\texttt{leaUserIncapable}</td>
<td>indicates the reason for leaving a chat is that a user has an old version of Skype which does not support multichat.</td>
</tr>
<tr>
<td>\texttt{leaAdderNotFriend}</td>
<td>indicates the reason for leaving a chat is that the recipient accepts messages from contacts only and the sender is not in this contact list.</td>
</tr>
<tr>
<td>\texttt{leaAddedNotAuthorized}</td>
<td>indicates the reason for leaving a chat is the recipient accepts authorised contacts only and has not authorised the sender.</td>
</tr>
<tr>
<td>\texttt{leaAddDeclined}</td>
<td>indicates that the invited user declined the chat invitation.</td>
</tr>
</tbody>
</table>

\textbf{Note:}

This message type existed in earlier versions of Skype, and is not used any more.
**leaUnsubscribe** indicates the reason for leaving is the user unsubscribed from the chat.

**enum TChatMemberRole**

Chat member roles.

**Enumerator:**

- `chatMemberRoleUnknown` Unknown role.
- `chatMemberRoleCreator` Chat creator.
- `chatMemberRoleMaster` Chat master.
- `chatMemberRoleHelper` Chat helper.
- `chatMemberRoleUser` Chat user.
- `chatMemberRoleListener` Chat listener.
- `chatMemberRoleApplicant` Chat applicant.

**enum TChatMessageStatus**

contains the status of a chat message.

**Enumerator:**
CMSUnknown indicates the status of the chat message is unknown.

cmsSending indicates the message is in the process of being sent.

cmsSent indicates the message has been sent.

cmsReceived indicates the message has been received.

cmsRead indicates the message has been read.

enum TChatMessageType

Chat message types.

Enumerator:

cmeUnknown Unknown message type.

cmeCreatedChatWith Chat created.

cmeSawMembers Saw chat members.

cmeAddedMembers Added chat members.

cmeSetTopic Chat topic changed.
<table>
<thead>
<tr>
<th>Event</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cmeSaid</td>
<td>Sent message.</td>
</tr>
<tr>
<td>cmeLeft</td>
<td>Left the chat.</td>
</tr>
<tr>
<td>cmeEmoted</td>
<td>Sent emoted message.</td>
</tr>
<tr>
<td>cmePostedContacts</td>
<td>Posted contacts.</td>
</tr>
<tr>
<td>cmeGapInChat</td>
<td>Indicates gap in chat.</td>
</tr>
<tr>
<td>cmeSetRole</td>
<td>Member role changed.</td>
</tr>
<tr>
<td>cmeKicked</td>
<td>Member kicked from chat.</td>
</tr>
<tr>
<td>cmeSetOptions</td>
<td>Chat options changed.</td>
</tr>
<tr>
<td>cmeKickBanned</td>
<td>Member kicked and banned from chat.</td>
</tr>
<tr>
<td>cmeJoinedAsApplicant</td>
<td>Member joined as applicant.</td>
</tr>
<tr>
<td>cmeSetPicture</td>
<td>Chat picture changed.</td>
</tr>
<tr>
<td>cmeSetGuidelines</td>
<td>Chat guidelines changed.</td>
</tr>
</tbody>
</table>

```
enum TChatMyStatus
```
Chat status.

**Enumerator:**

<table>
<thead>
<tr>
<th>Enumerated Status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>chatStatusUnknown</code></td>
<td>Unknown status.</td>
</tr>
<tr>
<td><code>chatStatusConnecting</code></td>
<td>Connecting to chat.</td>
</tr>
<tr>
<td><code>chatStatusWaitingRemoteAccept</code></td>
<td>Waiting for accept.</td>
</tr>
<tr>
<td><code>chatStatusAcceptRequired</code></td>
<td>Accept is required.</td>
</tr>
<tr>
<td><code>chatStatusPasswordRequired</code></td>
<td>Password is required.</td>
</tr>
<tr>
<td><code>chatStatusSubscribed</code></td>
<td>Subscribed to chat.</td>
</tr>
<tr>
<td><code>chatStatusUnsubscribed</code></td>
<td>Unsubscribed from chat.</td>
</tr>
<tr>
<td><code>chatStatusDisbanded</code></td>
<td>Chat was disbanded.</td>
</tr>
<tr>
<td><code>chatStatusQueuedBecauseChatIsFull</code></td>
<td>Queued because the chat is full.</td>
</tr>
<tr>
<td><code>chatStatusApplicationDenied</code></td>
<td>Application was denied.</td>
</tr>
<tr>
<td><code>chatStatusKicked</code></td>
<td>Kicked from chat.</td>
</tr>
<tr>
<td><code>chatStatusBanned</code></td>
<td>Banned from chat.</td>
</tr>
</tbody>
</table>
enum TChatOptions

Chat options.

**Enumerator:**

- **chatOptionJoiningEnabled**  
  Joining enabled.

- **chatOptionJoinersBecomeApplicants**  
  Joiners become applicants.

- **chatOptionJoinersBecomeListeners**  
  Joiners become listeners.

- **chatOptionHistoryDisclosed**  
  History disclosed.

- **chatOptionUsersAreListeners**  
  Users are listeners.

- **chatOptionTopicAndPictureLockedForUsers**  
  Topic and picture are locked for users.

enum TChatStatus

contains the status of a chat.
## Enumerator:

<table>
<thead>
<tr>
<th>Enum</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>chsUnknown</code></td>
<td>indicates the status of a chat is unknown.</td>
</tr>
<tr>
<td><code>chsLegacyDialog</code></td>
<td>indicates the chat is in a legacy window mode which does not support multichats.</td>
</tr>
<tr>
<td><code>chsDialog</code></td>
<td>indicates the chat is in a chat window.</td>
</tr>
<tr>
<td><code>chsMultiNeedAccept</code></td>
<td>indicates the chat is waiting for acceptance by multiple users.</td>
</tr>
<tr>
<td><code>chsMultiSubscribed</code></td>
<td>indicates the chat is subscribed to by multiple users.</td>
</tr>
<tr>
<td><code>chsUnsubscribed</code></td>
<td>indicates the chat is unsubscribed.</td>
</tr>
</tbody>
</table>

### enum TChatType

Chat types.

**Enumerator:**

<table>
<thead>
<tr>
<th>Enum</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>chatTypeUnknown</code></td>
<td>Unknown type.</td>
</tr>
<tr>
<td><code>chatTypeDialog</code></td>
<td>Chat with two participants.</td>
</tr>
<tr>
<td><code>chatTypeLegacyDialog</code></td>
<td>Legacy dialog.</td>
</tr>
</tbody>
</table>
chatTypeLegacyUnsubscribed  Legacy dialog (unsubscribed).

chatTypeMultiChat  Chat with multiple members.

chatTypeSharedGroup  Chat used to synchronize shared groups.

enum TConnectionStatus

indicates the status of a network connection.

**Enumerator:**

conUnknown  indicates that the status of a connection is unknown.

conOffline  indicates that a connection does not exist.

conConnecting  indicates that a connection is in the process of being established.

conPausing  indicates that a connection is pausing.

conOnline  indicates that a connection exists.

enum TFileTransferFailureReason

File transfer failure reason codes.
Enumerator:

`fileTransferFailureReasonSenderNotAuthorized`  
Sender is not authorized by remote user.

`fileTransferFailureReasonRemotelyCancelled`  
Remote user cancelled the file transfer.

`fileTransferFailureReasonFailedRead`  
File read error.

`fileTransferFailureReasonFailedRemoteRead`  
Remote file read error.

`fileTransferFailureReasonFailedWrite`  
File write error.

`fileTransferFailureReasonFailedRemoteWrite`  
Remote file write error.

`fileTransferFailureReasonRemoteDoesNotSupportFT`  
Remote party does not support file
Remote party has been offline too long.

**enum TFileTransferStatus**

File transfer status.

**Enumerator:**

- **fileTransferStatusNew**: A new file transfer.
- **fileTransferStatusConnecting**: Connecting to user.
- **fileTransferStatusWaitingForAccept**: Waiting user to accept the file transfer.
- **fileTransferStatusTransferring**: The file is being transferred.
- **fileTransferStatusTransferringOverRelay**: The file is being transferred over relay.
- **fileTransferStatusPaused**: The file transfer is paused.
enum TFileTransferType

File transfer types.

**Enumerator:**

- `fileTransferTypeIncoming`  Incoming file transfer.
- `fileTransferTypeOutgoing`  Outgoing file transfer.

enum TGroupType

contains the group type.

**Enumerator:**

- `grpUnknown`  indicates the group type is
<table>
<thead>
<tr>
<th>Group Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>grpCustomGroup</td>
<td>indicates the group type is user-defined.</td>
</tr>
<tr>
<td>grpAllUsers</td>
<td>indicates the group type is all other users a user knows about, including users in the contactlist, users recently contacted, and blocked users.</td>
</tr>
<tr>
<td>grpAllFriends</td>
<td>indicates the group type is all contacts in the user's contactlists.</td>
</tr>
<tr>
<td>grpSkypeFriends</td>
<td>indicates the group type is Skype contacts in the contactlist.</td>
</tr>
<tr>
<td>grpSkypeOutFriends</td>
<td>indicates the group type is SkypeOut contacts in the contactlist.</td>
</tr>
<tr>
<td>grpOnlineFriends</td>
<td>indicates the group type is Skype contacts in the contactlist who are online.</td>
</tr>
<tr>
<td>grpPendingAuthorizationFriends</td>
<td>indicates the group type is contacts in the contactlist who have not yet authorized the user.</td>
</tr>
<tr>
<td>Group Name</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>grpRecentlyContactedUsers</td>
<td>Indicates the group type is contacts the user has conversed with recently, including non-friends.</td>
</tr>
<tr>
<td>grpUsersWaitingMyAuthorization</td>
<td>Indicates the group type is contacts who are awaiting response to an authorisation request, including non-friends.</td>
</tr>
<tr>
<td>grpUsersAuthorizedByMe</td>
<td>Indicates the group type is all authorised contacts, including non-friends.</td>
</tr>
<tr>
<td>grpUsersBlockedByMe</td>
<td>Indicates the group type is all blocked contacts, including non-friends.</td>
</tr>
<tr>
<td>grpUngroupedFriends</td>
<td>Indicates the group type is all contacts in the contactlist that do not belong to any custom group.</td>
</tr>
<tr>
<td>grpSharedGroup</td>
<td>Indicates the group type is shared group.</td>
</tr>
<tr>
<td>grpProposedSharedGroup</td>
<td>Indicates the group type is proposed shared group.</td>
</tr>
</tbody>
</table>
enum TOnlineStatus

contains the online status of a user.

**Enumerator:**

- **olsUnknown** indicates the online status of the user is unknown.
- **olsOffline** indicates the user is offline (not connected).
- **olsOnline** indicates the user is online.
- **olsAway** indicates the user is away (has been inactive for certain period).
- **olsNotAvailable** indicates the user is not available.
- **olsDoNotDisturb** indicates the user is in do not disturb mode.
- **olsSkypeOut** indicates the user is in the SkypeOut contact list.
- **olsSkypeMe** indicates the user is in SkypeMe mode.

enum TPluginContactType

This property contains the contact type for the TPluginContext::pluginContextContact
**Enumerator:**

- `pluginContactTypeUnknown` This value indicates that the plug-in contact type is undefined.
- `pluginContactTypeAll` This value indicates that the plug-in is available for all contacts.
- `pluginContactTypeSkype` This value indicates that the plug-in is available for Skype contacts.
- `pluginContactTypeSkypeOut` This value indicates that the plug-in is available for SkypeOut contacts.

**enum TPluginContext**

contains the plug-in context.

**Enumerator:**

- `pluginContextUnknown` This value indicates that the plug-in context is undefined.
- `pluginContextChat` This value indicates that the plug-in works with chats.
- `pluginContextCall` This value indicates that the plug-in works with calls.
### $pluginContextContact$

works with contact list items.

### $pluginContextMyself$

This value indicates that the plug-in is a personal plug-in that works for a user who is logged into Skype.

### $pluginContextTools$

This value indicates that the plug-in is available as a generic tool.

### Enum $TSmsFailureReason$

This enumeration contains the reason an SMS message failed.

**Enumerator:**

- **$smsFailureReasonUnknown$**
  
  Indicates the SMS message failed for an unknown reason.

- **$smsFailureReasonMiscError$**
  
  Indicates the SMS message failed due to a miscellaneous error.

- **$smsFailureReasonServerConnectFailed$**
  
  Indicates a failure to connect to the server.

- **$smsFailureReasonUserNotAuthorized$**
  
  Indicates the user is not authorized.
<table>
<thead>
<tr>
<th>smsFailureReasonNoSmsCapability</th>
<th>has no SMS capability.</th>
</tr>
</thead>
<tbody>
<tr>
<td>smsFailureReasonInsufficientFunds</td>
<td>indicates the user does not have sufficient funds to send an SMS message.</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>smsFailureReasonInvalidConfirmationCode</td>
<td>indicates the SMS message has an invalid confirmation code.</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>smsFailureReasonUserBlocked</td>
<td>indicates the SMS message failed because the user is blocked by the SMS service.</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>smsFailureReasonIPBlocked</td>
<td>indicates the SMS message failed because the IP address is blocked by the SMS service.</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>smsFailureReasonNodeBlocked</td>
<td>indicates the SMS message failed because the node is blocked by the SMS service.</td>
</tr>
</tbody>
</table>
**enum TSmsMessageStatus**

contains the status of an SMS message.

**Enumerator:**

<table>
<thead>
<tr>
<th>Status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>smsMessageStatusUnknown</td>
<td>indicates the SMS message status is unknown.</td>
</tr>
<tr>
<td>smsMessageStatusReceived</td>
<td>indicates the SMS message was received.</td>
</tr>
<tr>
<td>smsMessageStatusRead</td>
<td>indicates the SMS message was read.</td>
</tr>
<tr>
<td>smsMessageStatusComposing</td>
<td>indicates the user is composing a SMS message.</td>
</tr>
<tr>
<td>smsMessageStatusSendingToServer</td>
<td>indicates that Skype is sending an SMS message to the server.</td>
</tr>
<tr>
<td>smsMessageStatusSentToServer</td>
<td>indicates an SMS message was sent to the server.</td>
</tr>
<tr>
<td>smsMessageStatusDelivered</td>
<td>indicates an SMS message was delivered.</td>
</tr>
</tbody>
</table>
**smsMessageStatusSomeTargetsFailed** indicates an SMS message failed to reach some targets.

**smsMessageStatusFailed** indicates an SMS message failed.

---

**enum TSmsMessageType**

contains the type of the SMS message.

**Enumerator:**

- **smsMessageTypeUnknown** indicates the SMS message is of an unknown type.
- **smsMessageTypeIncoming** indicates an incoming SMS message.
- **smsMessageTypeOutgoing** indicates an outgoing SMS message.
- **smsMessageTypeCCRequest** indicates the SMS message is a confirmation code request.
- **smsMessageTypeCCSubmit** indicates the SMS message is a confirmation code reply.

---

**enum TSmsTargetStatus**
contains the status of the target of an SMS message.

**Enumerator:**

<table>
<thead>
<tr>
<th>Status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>smsTargetStatusUnknown</code></td>
<td>Indicates the status of the target is unknown.</td>
</tr>
<tr>
<td><code>smsTargetStatusUndefined</code></td>
<td>Indicates the status of the target is undefined.</td>
</tr>
<tr>
<td><code>smsTargetStatusAnalyzing</code></td>
<td>Indicates that Skype is analyzing the target number.</td>
</tr>
<tr>
<td><code>smsTargetStatusAcceptable</code></td>
<td>Indicates the target number is acceptable.</td>
</tr>
<tr>
<td><code>smsTargetStatusNotRoutable</code></td>
<td>Indicates that there is no route to the target.</td>
</tr>
<tr>
<td><code>smsTargetStatusDeliveryPending</code></td>
<td>Indicates that delivery to the target is pending.</td>
</tr>
<tr>
<td><code>smsTargetStatusDeliverySuccessful</code></td>
<td>Indicates the message was delivered to the target.</td>
</tr>
<tr>
<td><code>smsTargetStatusDeliveryFailed</code></td>
<td>Indicates the SMS message failed to be delivered to the target.</td>
</tr>
</tbody>
</table>
enum TUserSex

contains information about the sex of a user.

**Enumerator:**

- *usexUnknown* indicates the sex of the user is unknown.
- *usexMale* indicates the user is male.
- *usexFemale* indicates the user is female.

enum TUserStatus

contains the Skype user status.

**Enumerator:**

- *cusUnknown* indicates the user status is unknown.
- *cusOffline* indicates the user is offline.
- *cusOnline* indicates the user is online.
- *cusAway* indicates the user is away.
- *cusNotAvailable* indicates the user is not available.
- *cusDoNotDisturb* indicates the user is in do not disturb mode.
`cusInvisible` indicates the user is invisible to others.

`cusLoggedOut` indicates the user is logged out.

`cusSkypeMe` indicates the user is in SkypeMe mode.

```
enum TVoicemailFailureReason
```

contains the reason for voicemail failure.

Voicemail failure reasons.

**Enumerator:**

- `vmrUnknown` indicates the reason for failure is unknown.
- `vmrNoError` indicates the reason for failure is none.
- `vmrMiscError` indicates the reason for failure is due to a miscellaneous error.
- `vmrConnectError` indicates the reason for failure is due to a connection error.
- `vmrNoPrivilege` indicates the reason for failure is the user has no voicemail service.
- `vmrNoVoicemail` indicates the reason for failure is there is no such voicemail.
<table>
<thead>
<tr>
<th>Enum Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>vmrFileReadError</td>
<td>Indicates the reason for failure is due to an error reading the file.</td>
</tr>
<tr>
<td>vmrFileWriteError</td>
<td>Indicates the reason for failure is due to an error writing the file.</td>
</tr>
<tr>
<td>vmrRecordingError</td>
<td>Indicates the reason for failure is due to a recording error.</td>
</tr>
<tr>
<td>vmrPlaybackError</td>
<td>Indicates the reason for failure is due to a playback error.</td>
</tr>
</tbody>
</table>

**Enum TVoicemailStatus**

Contains the status of a voicemail.

Voicemail status.

**Enumerator:**

<table>
<thead>
<tr>
<th>Enum Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>vmsUnknown</td>
<td>Indicates that voicemail status is unknown.</td>
</tr>
<tr>
<td>vmsNotDownloaded</td>
<td>Indicates that voicemail status is not played.</td>
</tr>
<tr>
<td>vmsDownloading</td>
<td>Indicates that voicemail status is in the process of downloading.</td>
</tr>
<tr>
<td>vmsUnplayed</td>
<td>Indicates that voicemail status is not downloaded.</td>
</tr>
<tr>
<td>Variable</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td><code>vmsBuffering</code></td>
<td>indicates that voicemail status is buffering.</td>
</tr>
<tr>
<td><code>vmsPlaying</code></td>
<td>indicates that voicemail status is playing.</td>
</tr>
<tr>
<td><code>vmsPlayed</code></td>
<td>indicates that voicemail status has been played.</td>
</tr>
<tr>
<td><code>vmsBlank</code></td>
<td>indicates that voicemail status is blank.</td>
</tr>
<tr>
<td><code>vmsRecording</code></td>
<td>indicates that voicemail status is recording.</td>
</tr>
<tr>
<td><code>vmsRecorded</code></td>
<td>indicates that voicemail status has been recorded.</td>
</tr>
<tr>
<td><code>vmsUploading</code></td>
<td>indicates that voicemail status is uploading.</td>
</tr>
<tr>
<td><code>vmsUploaded</code></td>
<td>indicates that voicemail status has been uploaded.</td>
</tr>
<tr>
<td><code>vmsDeleting</code></td>
<td>indicates that voicemail status is in the process of being deleted.</td>
</tr>
<tr>
<td><code>vmsFailed</code></td>
<td>indicates that voicemail status is failed.</td>
</tr>
</tbody>
</table>
**enum TVoicemailType**

defines voicemail types.

**Enumerator:**

<table>
<thead>
<tr>
<th>Enum</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>vmtUnknown</td>
<td>indicates the voicemail type is unknown.</td>
</tr>
<tr>
<td>vmtIncoming</td>
<td>indicates the voicemail type is incoming.</td>
</tr>
<tr>
<td>vmtDefaultGreeting</td>
<td>indicates the voicemail type is the default voicemail greeting.</td>
</tr>
<tr>
<td>vmtCustomGreeting</td>
<td>indicates the voicemail type is a custom voicemail greeting.</td>
</tr>
<tr>
<td>vmtOutgoing</td>
<td>indicates the voicemail type is an outgoing voicemail.</td>
</tr>
</tbody>
</table>
Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

<table>
<thead>
<tr>
<th>Class/Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>_ICallChannelManagerEvents</td>
<td>CallChannelManager event source interface</td>
</tr>
<tr>
<td>_ISkypeEvents</td>
<td>Skype event source interface</td>
</tr>
<tr>
<td>Application</td>
<td>This class enables two applications sharing the same name to communicate over the Skype network</td>
</tr>
<tr>
<td>ApplicationStream</td>
<td>This class manages application streams</td>
</tr>
<tr>
<td>ApplicationStreamCollection</td>
<td>This class manages application stream collections</td>
</tr>
<tr>
<td>Call</td>
<td>This class manages calls and call properties</td>
</tr>
<tr>
<td>CallChannel</td>
<td>This class forms a logical connection between the ApplicationStream and Call objects</td>
</tr>
<tr>
<td>CallChannelCollection</td>
<td>Container class for CallChannel objects</td>
</tr>
<tr>
<td>CallChannelManager</td>
<td>This class manages Call data channels</td>
</tr>
<tr>
<td>CallChannelMessage</td>
<td>This class manages CallChannel messages</td>
</tr>
<tr>
<td>CallCollection</td>
<td>This class manages Call collections</td>
</tr>
<tr>
<td>Chat</td>
<td>This class manages chats</td>
</tr>
<tr>
<td>ChatCollection</td>
<td>This class manages Chat collections</td>
</tr>
<tr>
<td>Class</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>ChatMember</td>
<td>This class manages chat member</td>
</tr>
<tr>
<td>ChatMemberCollection</td>
<td>This class manages chat member collections</td>
</tr>
<tr>
<td>ChatMessage</td>
<td>This class manages chat messages</td>
</tr>
<tr>
<td>ChatMessageCollection</td>
<td>This class manages chat message collections</td>
</tr>
<tr>
<td>Client</td>
<td>This class controls features related to the Skype user interface</td>
</tr>
<tr>
<td>Command</td>
<td>This class manages Skype API command objects</td>
</tr>
<tr>
<td>Conference</td>
<td>This class manages conference calls</td>
</tr>
<tr>
<td>ConferenceCollection</td>
<td>This class manages conference collections</td>
</tr>
<tr>
<td>Conversion</td>
<td>This class converts text to constant values and converts constant values to text</td>
</tr>
<tr>
<td>Group</td>
<td>This class manages contact groups</td>
</tr>
<tr>
<td>GroupCollection</td>
<td>This class manages collections of Group objects</td>
</tr>
<tr>
<td>IApplication</td>
<td>Application class default interface</td>
</tr>
<tr>
<td>IApplicationStream</td>
<td>ApplicationStream class default interface</td>
</tr>
<tr>
<td>IApplicationStreamCollection</td>
<td>ApplicationStreamCollection class default interface</td>
</tr>
<tr>
<td>ICall</td>
<td>Call class default interface</td>
</tr>
<tr>
<td>ICallChannel</td>
<td>CallChannel class default interface</td>
</tr>
<tr>
<td>CallChannelCollection</td>
<td>CallChannelCollection class default interface</td>
</tr>
<tr>
<td>Interface Name</td>
<td>Class Description</td>
</tr>
<tr>
<td>--------------------------------------</td>
<td>-------------------------------------------------------------</td>
</tr>
<tr>
<td>ICallChannelCollection</td>
<td>default interface</td>
</tr>
<tr>
<td>ICallChannelManager</td>
<td>CallChannelManager class default interface</td>
</tr>
<tr>
<td>ICallChannelMessage</td>
<td>CallChannelMessage class default interface</td>
</tr>
<tr>
<td>ICallCollection</td>
<td>CallCollection class default interface</td>
</tr>
<tr>
<td>IChat</td>
<td>Chat class default interface</td>
</tr>
<tr>
<td>IChatCollection</td>
<td>ChatCollection class default interface</td>
</tr>
<tr>
<td>IChatMember</td>
<td>ChatMember class default interface</td>
</tr>
<tr>
<td>IChatMemberCollection</td>
<td>ChatMemberCollection class default interface</td>
</tr>
<tr>
<td>IChatMessage</td>
<td>ChatMessage class default interface</td>
</tr>
<tr>
<td>IChatMessageCollection</td>
<td>ChatMessageCollection class default interface</td>
</tr>
<tr>
<td>IClient</td>
<td>Client class default interface</td>
</tr>
<tr>
<td>ICommand</td>
<td>Command class default interface</td>
</tr>
<tr>
<td>IConference</td>
<td>Conference class default interface</td>
</tr>
<tr>
<td>IConferenceCollection</td>
<td>ConferenceCollection class default interface</td>
</tr>
<tr>
<td>IConversion</td>
<td>Conversion class default interface</td>
</tr>
<tr>
<td>IFileTransfer</td>
<td>File transfer object default interface</td>
</tr>
<tr>
<td>IFileTransferCollection</td>
<td>File transfer collection class default interface</td>
</tr>
<tr>
<td>IGroup</td>
<td>Group class default interface</td>
</tr>
<tr>
<td>IGroupCollection</td>
<td>GroupCollection class default interface</td>
</tr>
<tr>
<td>Interface/Collection</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>IGroupCollection</td>
<td>interface</td>
</tr>
<tr>
<td>IParticipant</td>
<td>Participant class default interface</td>
</tr>
<tr>
<td>IParticipantCollection</td>
<td>ParticipantCollection class default interface</td>
</tr>
<tr>
<td>IPluginEvent</td>
<td>This is the default interface for the PluginEvent class</td>
</tr>
<tr>
<td>IPluginMenuItem</td>
<td>Class PluginMenuItem default interface</td>
</tr>
<tr>
<td>IProfile</td>
<td>Profile class default interface</td>
</tr>
<tr>
<td>ISettings</td>
<td>Settings class default interface</td>
</tr>
<tr>
<td>ISkype</td>
<td>Skype is the default interface for the class</td>
</tr>
<tr>
<td>ISmsChunk</td>
<td>SmsChunk is the default interface for this class</td>
</tr>
<tr>
<td>ISmsChunkCollection</td>
<td>SmsChunkCollection is the default interface for this class</td>
</tr>
<tr>
<td>ISmsMessage</td>
<td>SmsMessage is the default interface for this class</td>
</tr>
<tr>
<td>ISmsMessageCollection</td>
<td>SmsMessageCollection is the default interface for this class</td>
</tr>
<tr>
<td>ISmsTarget</td>
<td>SmsTarget is the default interface for this class</td>
</tr>
<tr>
<td>ISmsTargetCollection</td>
<td>SmsTargetCollection is the default interface for this class</td>
</tr>
<tr>
<td>IUser</td>
<td>User class default interface. This interface enables you to read user properties</td>
</tr>
<tr>
<td>IUserCollection</td>
<td>UserCollection class default interface</td>
</tr>
<tr>
<td>IVoicemail</td>
<td>Voicemail class default interface</td>
</tr>
<tr>
<td>IVoicemailCollection</td>
<td>VoicemailCollection class</td>
</tr>
<tr>
<td>Class</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>---------------------------------------------------------------</td>
</tr>
<tr>
<td>Participant</td>
<td>This class manages participants in a conference call</td>
</tr>
<tr>
<td>ParticipantCollection</td>
<td>This class manages collections of participants in a conference call</td>
</tr>
<tr>
<td>PluginEvent</td>
<td>This class manages plug-in event</td>
</tr>
<tr>
<td>PluginMenuItem</td>
<td>Plugin menu item</td>
</tr>
<tr>
<td>Profile</td>
<td>This class manages the user profile</td>
</tr>
<tr>
<td>Settings</td>
<td>This class manages the Skype API general parameters</td>
</tr>
<tr>
<td>Skype</td>
<td>Skype API COM wrapper</td>
</tr>
<tr>
<td>SmsChunk</td>
<td>SMS message chunk</td>
</tr>
<tr>
<td>SmsChunkCollection</td>
<td>Collection class for SmsChunk objects</td>
</tr>
<tr>
<td>SmsMessage</td>
<td>SMS message</td>
</tr>
<tr>
<td>SmsMessageCollection</td>
<td>Collection class for SMS messages</td>
</tr>
<tr>
<td>SmsTarget</td>
<td>SMS message target</td>
</tr>
<tr>
<td>SmsTargetCollection</td>
<td>Collection class for SMS targets</td>
</tr>
<tr>
<td>User</td>
<td>This class manages user properties</td>
</tr>
<tr>
<td>UserCollection</td>
<td>User collection</td>
</tr>
<tr>
<td>Voicemail</td>
<td>This class manages Skype voicemail</td>
</tr>
<tr>
<td>VoicemailCollection</td>
<td>Voicemail collection</td>
</tr>
</tbody>
</table>
Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Public Member Functions
CallChannelManager event source interface. More...

import "Skype4COM.idl";

List of all members.
# Public Member Functions

<table>
<thead>
<tr>
<th>HRESULT</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HRESULT</td>
<td><strong>Channels</strong></td>
<td>([in] ICallChannelManager *pManager,[in] ICallChannelCollection *pChannels)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This event indicates a change in the call channels.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>Message</strong></td>
<td>([in] ICallChannelManager *pManager,[in] ICallChannel *pChannel,[in] ICallChannelMessage *pMessage)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This event indicates that a call channel message has been received.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>Created</strong></td>
<td>()</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This event indicates that the Application context was successfully created.</td>
</tr>
</tbody>
</table>
Detailed Description

CallChannelManager event source interface.
**Member Function Documentation**

**HRESULT Channels ([in] ICallChannelManager * pManager, [in] ICallChannelCollection * pChannels )**

This event indicates a change in the call channels.

**Parameters:**

- **[in] pManager** contains the interface pointer for the `CallChannelManager` object.
- **[in] pChannels** contains the interface pointer for the `CallChannelCollection` object.

**HRESULT Message ([in] ICallChannelManager * pManager, [in] ICallChannel * pChannel, [in] ICallChannelMessage * pMessage )**

This event indicates that a call channel message has been received.

**Parameters:**

- **[in] pManager** contains the interface pointer for the `CallChannelManager` object.
- **[in] pChannel** contains the interface pointer for the `CallChannel` object.
- **[in] pMessage** contains the interface pointer for the `CallChannelMessage` object.

The documentation for this interface was generated from the following file:

- **Skype4COM.idl**
Copyright © 2006 Skype Limited. All rights reserved.
## Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Public Member Functions
Skype event source interface. More...

import "Skype4COM.idl";

List of all members.
### Public Member Functions

<table>
<thead>
<tr>
<th>HRESULT</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HRESULT</td>
<td><strong>Command</strong> ([in] ICommand *pCommand)</td>
<td>This event indicates the command object to use with the Skype API.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>Reply</strong> ([in] ICommand *pCommand)</td>
<td>This event indicates the API response to the command object.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>Error</strong> ([in] ICommand *pCommand,[in] LONG Number,[in] BSTR Description)</td>
<td>This event indicates the error number and description associated with a bad command object.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>AttachmentStatus</strong> ([in] TAttachmentStatus Status)</td>
<td>This event is caused by a change in the status of an attachment to the Skype API.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>ConnectionStatus</strong> ([in] TConnectionStatus Status)</td>
<td>This event is caused by a connection status change.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>UserStatus</strong> ([in] TUserStatus Status)</td>
<td>This event is caused by a user status change.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>OnlineStatus</strong> ([in] IUser *pUser,[in] TOnlineStatus Status)</td>
<td>This event is caused by a change in the online status of a user.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>CallStatus</strong> ([in] ICall *pCall,[in] TCallStatus Status)</td>
<td>This event is caused by a change in call status.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>CallHistory</strong> ()</td>
<td>This event is caused by a change in call history.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>Mute</strong> ([in] VARIANT_BOOL Mute)</td>
<td>This event is caused by a change in mute status.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>MessageStatus</strong> ([in] IChatMessage *pMessage,[in] TChatMessageStatus Status)</td>
<td>This event is caused by a change in chat message status.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>MessageHistory</strong> ([in] BSTR Username)</td>
<td></td>
</tr>
</tbody>
</table>
This event is caused by a change in message history.

<table>
<thead>
<tr>
<th>HRESULT</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HRESULT</td>
<td><strong>AutoAway</strong> ([in] VARIANT_BOOL Automatic)</td>
<td>This event is caused by a change of auto away status.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>CallDtmfReceived</strong> ([in] ICall *pCall,[in] BSTR Code)</td>
<td>This event is caused by a call DTMF event.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>VoicemailStatus</strong> ([in] IVoicemail *pMail,[in] TVoicemailStatus Status)</td>
<td>This event is caused by a change in voicemail status.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>ApplicationConnecting</strong> ([in] IApplication *pApp,[in] IUserCollection *pUsers)</td>
<td>This event is caused by users connecting to an application.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>ApplicationStreams</strong> ([in] IApplication *pApp,[in] IApplicationStreamCollection *pStreams)</td>
<td>This event is caused by a change in application streams.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>ApplicationDatagram</strong> ([in] IApplication *pApp,[in] IApplicationStream *pStream,[in] BSTR Text)</td>
<td>This event is caused by the arrival of an application datagram.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>ApplicationSending</strong> ([in] IApplication *pApp,[in] IApplicationStreamCollection *pStreams)</td>
<td>This event is caused by a change of application sending streams.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>ApplicationReceiving</strong> ([in] IApplication *pApp,[in] IApplicationStreamCollection *pStreams)</td>
<td>This event is caused by a change of application receiving streams.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>ContactsFocused</strong> ([in] BSTR Username)</td>
<td>This event is caused by the contacts tab gaining or losing focus.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>GroupVisible</strong> ([in] IGroup *pGroup,[in] VARIANT_BOOL Visible)</td>
<td>This event is caused by a user hiding/showing a group in the contacts tab.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>GroupExpanded</strong> ([in] IGroup *pGroup,[in] VARIANT_BOOL Expanded)</td>
<td></td>
</tr>
</tbody>
</table>
This event is caused by a user expanding or collapsing a group in the contacts tab.

<table>
<thead>
<tr>
<th>HRESULT</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HRESULT</td>
<td><strong>GroupUsers</strong> ([in] IGroup *pGroup,[in] IUserCollection *pUsers)</td>
<td>This event is caused by a change in a contact group.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>GroupDeleted</strong> ([in] LONG GroupId)</td>
<td>This event is caused by a user deleting a custom contact group.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>UserMood</strong> ([in] IUser *pUser,[in] BSTR MoodText)</td>
<td>This event is caused by a change in the mood text of the user.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>SmsMessageStatusChanged</strong> ([in] ISmsMessage *pMessage,[in] TSmsMessageStatus Status)</td>
<td>This event is caused by a change in the SMS message status.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>SmsTargetStatusChanged</strong> ([in] ISmsTarget *pTarget,[in] TSmsTargetStatus Status)</td>
<td>This event is caused by a change in the SMS target status.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>CallInputStatusChanged</strong> ([in] ICall *pCall,[in] VARIANT_BOOL Status)</td>
<td>This event is caused by a change in the Call voice input status change.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>AsyncSearchUsersFinished</strong> ([in] LONG Cookie,[in] IUserCollection *pUsers)</td>
<td>This event occurs when a search is completed.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>CallSeenStatusChanged</strong> ([in] ICall *pCall,VARIANT_BOOL Status)</td>
<td>This event occurs when the seen status of a call changes.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>PluginEventClicked</strong> ([in] IPluginEvent *pEvent)</td>
<td>This event occurs when a user clicks on a plug-in event.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>PluginMenuItemClicked</strong> ([in] IPluginMenuItem *pMenuItem,[in] IUserCollection *pUsers,[in] TPluginContext PluginContext,[in] BSTR ContextId)</td>
<td>This event occurs when a user clicks on a menu item.</td>
</tr>
<tr>
<td>HRESULT</td>
<td>Event Description</td>
<td>Details</td>
</tr>
<tr>
<td>---------</td>
<td>-------------------</td>
<td>---------</td>
</tr>
<tr>
<td>HRESULT</td>
<td>WallpaperChanged</td>
<td>when a wallpaper changes.</td>
</tr>
<tr>
<td>HRESULT</td>
<td>FileTransferStatusChanged</td>
<td>when a file transfer status changes.</td>
</tr>
<tr>
<td>HRESULT</td>
<td>CallTransferStatusChanged</td>
<td>when a call transfer status changes.</td>
</tr>
<tr>
<td>HRESULT</td>
<td>ChatMembersChanged</td>
<td>when chat members change.</td>
</tr>
<tr>
<td>HRESULT</td>
<td>ChatMemberRoleChanged</td>
<td>when a chat member role changes.</td>
</tr>
<tr>
<td>HRESULT</td>
<td>CallVideoStatusChanged</td>
<td>when a call video status changes.</td>
</tr>
<tr>
<td>HRESULT</td>
<td>CallVideoSendStatusChanged</td>
<td>when a call video send status changes.</td>
</tr>
<tr>
<td>HRESULT</td>
<td>CallVideoReceiveStatusChanged</td>
<td>when a call video receive status changes.</td>
</tr>
<tr>
<td>HRESULT</td>
<td>SilentModeStatusChanged</td>
<td>when a silent mode is switched off.</td>
</tr>
<tr>
<td>HRESULT</td>
<td>UILanguageChanged</td>
<td>when user changes Skype client language.</td>
</tr>
<tr>
<td>HRESULT</td>
<td>UserAuthorizationRequestReceived</td>
<td>when user sends you authorization request.</td>
</tr>
</tbody>
</table>
Detailed Description

**Skype** event source interface.

**Examples:**

- `Call.php`
Member Function Documentation

HRESULT ApplicationConnecting ( [in] IApplication * pApp,
    [in] IUserCollection * pUsers
 )

This event is caused by users connecting to an application.

Parameters:

HRESULT ApplicationDatagram ( [in] IApplication * pApp,
    [in] IApplicationStream * pStream
    [in] BSTR Text
 )

This event is caused by the arrival of an application datagram.

Parameters:
    [in] Text contains the datagram text.

HRESULT ApplicationReceiving ( [in] IApplication * pApp,
    [in] IApplicationStreamCollection
 )

This event is caused by a change of application receiving streams.
**Parameters:**

- `[in] pApp` contains the interface pointer for the `Application` object.
- `[in] pStreams` contains the interface pointer for the `ApplicationStreamCollection` object.

**HRESULT ApplicationSending ([in] IApplication * [in] IApplicationStreamCollection *)**

This event is caused by a change of application sending streams.

**Parameters:**

- `[in] pApp` contains the interface pointer for the `Application` object.
- `[in] pStreams` contains the interface pointer for the `ApplicationStreamCollection` object.

**HRESULT ApplicationStreams ([in] IApplication * [in] IApplicationStreamCollection *)**

This event is caused by a change in application streams.

**Parameters:**

- `[in] pApp` contains the interface pointer for the `Application` object.
- `[in] pStreams` contains the interface pointer for the `ApplicationStreamCollection` object.


...
This event occurs when a search is completed.

**Parameters:**

- **Cookie** contains the search identifier.
- **pUsers** contains the interface pointer for the **UserCollection** object.

See also:

**ISkype::AsyncSearchUsers**

**HRESULT AttachmentStatus ([in] TAttachmentStatus Status)**

This event is caused by a change in the status of an attachment to the Skype API.

**Parameters:**

- **Status** contains the new attachment status.

**Examples:**

**Call.php.**

**HRESULT AutoAway ([in] VARIANT_BOOL Automatic)**

This event is caused by a change of auto away status.

**Parameters:**

- **Automatic** contains the new auto away status.

**HRESULT CallDtmfReceived ([in] ICall * pCall,**

- **[in] BSTR Code**

This event is caused by a call DTMF event.
Parameters:

[in] pCall contains the interface pointer for the Call object.
[in] Code contains the received DTMF code.

Note:
This event can be received with P2P calls.

HRESULT CallInputStatusChanged ([in] ICall * pCall,
[in] VARIANT_BOOL Status)

This event is caused by a change in the Call voice input status change.

Parameters:

[in] pCall contains the interface pointer for the Call object.
[in] Status contains the voice input status (VARIANT_TRUE: Active, VARIANT_FALSE: Inactive).

See also:
ICall::InputDevice

HRESULT CallSeenStatusChanged ([in] ICall * pCall,
VARIANT_BOOL Status)

This event occurs when the seen status of a call changes.

Parameters:

[in] pCall contains the interface pointer for the Call object.

See also:
ICall::Seen

HRESULT CallStatus ([in] ICall * pCall, [in] TCallStatus Status)

This event is caused by a change in call status.

Parameters:
[in] pCall receives the interface pointer for the Call object.
[in] Status contains the new call status.

Examples:
Call.php.

HRESULT CallTransferStatusChanged ([in] ICall * pCall, [in] TCallStatus Status)

This event occurs when a call transfer status changes.

Parameters:
[in] pCall contains the interface pointer for the Call object.
[in] Status contains the new call transfer status.

HRESULT CallVideoReceiveStatusChanged ([in] ICall * pCall, [in] TCallVideoSendStatus Status)

This event occurs when a call video receive status changes.

Parameters:
[in] pCall contains the interface pointer for the Call object.
[in] **Status** contains new video receive status.

 HRESULT CallVideoSendStatusChanged ( [in] ICall * pCall ,
                                     [in] TCallVideoSendStatus Status )

This event occurs when a call video send status changes.

**Parameters:**
- [in] **pCall** contains the interface pointer for the Call object.
- [in] **Status** contains new video send status.

 HRESULT CallVideoStatusChanged ( [in] ICall * pCall ,
                                     [in] TCallVideoStatus Status )

This event occurs when a call video status changes.

**Parameters:**
- [in] **pCall** contains the interface pointer for the Call object.
- [in] **Status** contains new video status.

 HRESULT ChatMemberRoleChanged ( [in] IChatMember * pMember ,
                                 [in] TChatMemberRole Role )

This event occurs when a chat member role changes.

**Parameters:**
- [in] **pMember** contains the interface pointer for the ChatMember object.
- [in] **Role** contains the new role code.
HRESULT ChatMembersChanged ([in] IChat * pChat, [in] IUserCollection * pMembers )

This event occurs when a chat members change.

Parameters:
[in] pMembers contains the interface pointer for the UserCollection object.

HRESULT Command ([in] ICommand * pCommand )

This event indicates the command object to use with the Skype API.

Parameters:

HRESULT ConnectionStatus ([in] TConnectionStatus Status )

This event is caused by a connection status change.

Parameters:
[in] Status contains the new connection status.

HRESULT ContactsFocused ([in] BSTR Username )

This event is caused by the contacts tab gaining or losing focus.

Parameters:
[in] Username is an optional field to enter the Skypename of the user.
HRESULT Error ([in] ICommand * pCommand,  
   [in] LONG Number,  
   [in] BSTR Description
)

This event indicates the error number and description associated with a bad command object.

**Parameters:**

- [in] **pCommand** contains the **Command** object.
- [in] **Number** contains the error number to be returned by the Skype API.
- [in] **Description** contains a description of the error.

HRESULT FileTransferStatusChanged ([in] IFileTransfer * pTransfer,  
   [in] TFileTransferStatus Status
)

This event occurs when a file transfer status changes.

**Parameters:**

- [in] **pTransfer** contains the interface pointer for the File Transfer object.
- [in] **Status** contains the new file transfer status.

**See also:**

- ISkype::FileTransfers, ISkype::ActiveFileTransfers

HRESULT GroupDeleted ([in] LONG GroupId)

This event is caused by a user deleting a custom contact group.

**Parameters:**
[in] `GroupId` contains the ID of the deleted group.

```csharp
HRESULT GroupExpanded ([in] IGroup * pGroup,
                      [in] VARIANT_BOOL Expanded)
```

This event is caused by a user expanding or collapsing a group in the contacts tab.

**Parameters:**
- [in] `pGroup` contains the interface pointer for the `Group` object.
- [in] `Expanded` is a Boolean indicator. When the value is set to True, the group is expanded.

```csharp
HRESULT GroupUsers ([in] IGroup * pGroup,
                     [in] IUserCollection * pUsers)
```

This event is caused by a change in a contact group.

**Parameters:**
- [in] `pGroup` contains the interface pointer for the `Group` object.
- [in] `pUsers` contains the interface pointer for the `UserCollection` object.

```csharp
HRESULT GroupVisible ([in] IGroup * pGroup,
                      [in] VARIANT_BOOL Visible)
```

This event is caused by a user hiding/showing a group in the contacts tab.
Parameters:

- `[in] pGroup` contains the interface pointer for the `Group` object.
- `[in] Visible` is a Boolean indicator. It is set to `True` when the group is visible in the contacts tab.

**HRESULT MessageHistory ([in] BSTR Username)**

This event is caused by a change in message history.

**Parameters:**

- `[in] Username` contains the name of the user whose message history changed.


This event is caused by a change in chat message status.

**Parameters:**

- `[in] pMessage` receives the interface pointer for the `ChatMessage` object.
- `[in] Status` contains the new message status.

**HRESULT Mute ([in] VARIANT_BOOL Mute)**

This event is caused by a change in mute status.

**Parameters:**

- `[in] Mute` contains the new mute status value.

**HRESULT OnlineStatus ([in] IUser * pUser)**
This event is caused by a change in the online status of a user.

**Parameters:**
- `[in]` **pUser** receives the interface pointer for the **User** object.
- `[in]` **Status** contains the new online status of the user.

**HRESULT PluginEventClicked ([in] IPluginEvent * pEvent )**

This event occurs when a user clicks on a plug-in event.

**Parameters:**
- `[in]` **pEvent** contains the interface pointer for the **PluginEvent** object.

**See also:**
- IPluginEvent


This event occurs when a user clicks on a menu item.

**Parameters:**
- `[in]` **pMenuItem** contains the interface pointer for the **PluginMenuItem** object.
- `[in]` **pUsers** contains the interface pointer for the **UserCollection** object.
[in] **PluginContext** contains the plug-in context.

[in] **ContextId** contains the context id.

See also:

IPluginMenuItem

**HRESULT Reply ([in] ICommand * pCommand)**

This event indicates the API response to the command object.

**Parameters:**

[in] **pCommand** contains the response to the **Command** object.

**HRESULT SilentModeStatusChanged ([in] VARIANT_BOOL Silent)**

This event occurs when a silent mode is switched off.

**Parameters:**

[in] **Silent** contains the Skype client silent status.

**HRESULT SmsMessageStatusChanged ([in] ISmsMessage * pMessage [in] TSmsMessageStatus)**

This event is caused by a change in the SMS message status.

**Parameters:**

[in] **pMessage** contains the interface pointer for the **SmsMessage** object.

[in] **Status** contains the new message status.

**HRESULT SmsTargetStatusChanged ([in] ISmsTarget * pTarget)**
This event is caused by a change in the SMS target status.

**Parameters:**
- `[in] pTarget` contains the interface pointer for the `SmsTarget` object.
- `[in] Status` contains the new target status.

```csharp
HRESULT UILanguageChanged ([in] BSTR Code)
```

This event occurs when user changes Skype client language.

**Parameters:**

```csharp
HRESULT UserAuthorizationRequestReceived ([in] IUser * pUser)
```

This event occurs when user sends you authorization request.

**Parameters:**
- `[in] pUser` contains the interface pointer for the `User` object.

```csharp
```

This event is caused by a change in the mood text of the user.

**Parameters:**
- `[in] pUser` contains the interface pointer for the `User` object.
[in] **MoodText** contains the new mood text.

**HRESULT UserStatus** ([in] **TUserStatus** Status)

This event is caused by a user status change.

**Parameters:**
- [in] **Status** contains the new user status.

**HRESULT VoicemailStatus** ([in] **IVoicemail** pMail, [in] **TVoicemailStatus** Status)

This event is caused by a change in voicemail status.

**Parameters:**
- [in] **pMail** contains the interface pointer for the Voicemail object.
- [in] **Status** contains the new voicemail status.

**HRESULT WallpaperChanged** ([in] **BSTR** Path)

This event occurs when a wallpaper changes.

**Parameters:**
- [in] **Path** to new wallpaper bitmap.

**See also:**
IClient::Wallpaper

The documentation for this interface was generated from the following file:
Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SKYPE4COMLib</td>
<td>Application</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
This class enables two applications sharing the same name to communicate over the Skype network. More...

import "Skype4COM.idl";
Detailed Description

This class enables two applications sharing the same name to communicate over the Skype network.

See also:
   IApplication, IApplicationStream

The documentation for this class was generated from the following file:

- Skype4COM.idl

Copyright © 2006 Skype Limited. All rights reserved.
<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td>Modules</td>
<td>Classes</td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td>SKYPE4COMLib</td>
<td>ApplicationStream</td>
</tr>
</tbody>
</table>
ApplicationStream Class Reference

This class manages application streams. More...

import "Skype4COM.idl";
Detailed Description

This class manages application streams.

The ApplicationStream class provides a reliable connection between two application endpoints.

See also:
   IApplicationStream

The documentation for this class was generated from the following file:

- Skype4COM.idl

Copyright © 2006 Skype Limited. All rights reserved.
# Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Files</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class List</td>
<td>Class Members</td>
</tr>
</tbody>
</table>

| SKYPE4COMLib | ApplicationStreamCollection |
This class manages application stream collections. 

import "Skype4COM.idl";
Detailed Description

This class manages application stream collections.

See also:
IAplicationStreamCollection, IApplicationStream

The documentation for this class was generated from the following file:

- Skype4COM.idl

Copyright © 2006 Skype Limited. All rights reserved.
Call Class Reference

This class manages calls and call properties. More...

import "Skype4COM.idl";
Detailed Description

This class manages calls and call properties.

Use the Call class to manage call properties, answer incoming calls, change call status, and send DTMF tones.

Example 1: Placing a call

```vbs
'// Create a Skype object:
Set oSkype = WScript.CreateObject("Skype4COM.Skype", "Skype_")

'// Place a call using this Skype object:
Set oCall = oSkype.PlaceCall("echo123")

'// To place a call on hold:
oCall.Hold

'// To resume a call that was on hold:
oCall.Resume

'// To end a call:
oCall.Finish

'// Call status events:
Public Sub Skype_CallStatus(ByRef oCall, ByVal Status)
End Sub
```

Example 2: Accessing call properties

```vbs
'// To query a call object with a specified ID:
Set oCall = oSkype.Call(oId)

'// To query the Skypename of a remote caller:
```
Set oHandle = oCall.PartnerHandle

'// To query the display name of a remote caller:
Set oName = oCall.PartnerDisplayName

See also:
ICall, ICallCollection, _ISkypeEvents

The documentation for this class was generated from the following file:

- Skype4COM.idl

Copyright © 2006 Skype Limited. All rights reserved.
Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

SKYPE4COMLib CallChannel
CallChannel Class Reference

This class forms a logical connection between the ApplicationStream and Call objects. More...

import "Skype4COM.idl";
Detailed Description

This class forms a logical connection between the ApplicationStream and Call objects.

The CallChannel is an abstract layer above the Call and ApplicationStream objects. The lifetime of the CallChannel (and the corresponding ApplicationStream) is determined by the lifetime of the active Call.

See also:

ICallChannel, ICallChannelCollection, ICallChannelManager

The documentation for this class was generated from the following file:

- Skype4COM.idl

Copyright Ⓒ 2006 Skype Limited. All rights reserved.
<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td>Modules</td>
<td>Classes</td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**SKYPE4COMLib**

**CallChannelCollection**
CallChannelCollection Class Reference

Container class for CallChannel objects. More...

import "Skype4COM.idl";
Detailed Description

Container class for CallChannel objects.

See also:
ICallChannel, ICallChannelCollection, ICallChannelManager

The documentation for this class was generated from the following file:

- Skype4COM.idl

Copyright © 2006 Skype Limited. All rights reserved.
<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**SKYPE4COMLib** | **CallChannelManager**
CallChannelManager Class Reference

This class manages Call data channels. More...

import "Skype4COM.idl";
Detailed Description

This class manages Call data channels.

The CallChannelManager class listens for incoming Skype Call and attempts to create a data channel between the callers. The ApplicationStream is created when the ICall::Status is TCallStatus::clsRinging. The ApplicationStream is deleted when the call status changes to TCallStatus::clsCancelled, TCallStatus::clsFailed, TCallStatus::clsFinished or TCallStatus::clsRefused.

**Note:**
To create a data channel it is necessary to have an instance of the CallChannelManager with the same name running on both endpoints of a call. It is possible to have multiple data channels running at the same time.

See also:
ICallChannelManager, _ICallChannelManagerEvents, ICallChannel

Example:

```vbnet
'// To create a channel manager object:
Set oMgr = WScript.CreateObject("Skype4COM.CallChannelManager", "ChannelManager_")

'// Create the application context:
oMgr.Create("TestApplicationName")

'// Wait for the application context to be created:
Do While Not oMgr.Created
    WScript.Sleep(1000)
Loop

'// This event occurs when a new channel is c
```
reated or an existing channel is deleted:
Public Sub ChannelManager_Channels(ByVal aManager As Object, ByVal aChannels As Object)
End Sub

'// This event occurs when a message arrives from a remote user
Public Sub ChannelManager_Message(ByVal aManager As Object, ByVal aChannel As Object, ByVal aMessage As Object)
End Sub

'// This event occurs when an application context is created
Public Sub ChannelManager_Created()
End Sub

The documentation for this class was generated from the following file:

- Skype4COM.idl

Copyright © 2006 Skype Limited. All rights reserved.
## Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td>Modules</td>
<td>Classes</td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**SKYPE4COMLib** | **CallChannelMessage**
CallChannelMessage Class Reference

This class manages CallChannel messages. More...

import "Skype4COM.idl";
Detailed Description

This class manages CallChannel messages.

See also: 
ICallChannelMessage, ICallChannelManager

The documentation for this class was generated from the following file:

- Skype4COM.idl

Copyright © 2006 Skype Limited. All rights reserved.
# Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SKYPE4COMLib</th>
<th>CallCollection</th>
</tr>
</thead>
</table>
CallCollection Class Reference

This class manages Call collections. More...

import "Skype4COM.idl";
Detailed Description

This class manages **Call** collections.

The **CallCollection** class is used to manage call collections. To query items in a collection, use **For Each .. Next** statements.

Example:

```csharp
'// To query call history:
For Each oCall In oSkype.Calls
Next

'// To query the call history for a user:
For Each oCall In oSkype.Calls("echo123")
Next

'// To query active calls:
For Each oCall In oSkype.ActiveCalls
Next

'// To query missed calls:
For Each oCall In oSkype.MissedCalls
Next
```

See also:

**ICall, ICallCollection**

The documentation for this class was generated from the following file:

- **Skype4COM.idl**

---

Copyright � 2006 **Skype Limited**. All rights reserved.
## Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

SKYPE4COM Lib  Chat
Chat Class Reference

This class manages chats. More...

import "Skype4COM.idl";
Detailed Description

This class manages chats.

The **Chat** class groups messages with the same chat name.

**Example:**

```vbnet
' // To query a chat object with a specified name:
Set oChat = oSkype.Chat(oName)

' // To query a chat topic:
Set oTopic = oChat.Topic

' // To query a chat timestamp:
Set oTime = oChat.Timestamp

' // To iterate through messages in a chat:
For Each oMsg In oChat.Messages
Next
```

See also:

*IChat, IChatCollection*

---

The documentation for this class was generated from the following file:

- Skype4COM.idl

---

Copyright © 2006 Skype Limited. All rights reserved.
# Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td>Modules</td>
<td>Classes</td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**SKYPE4COMLib**

**ChatCollection**
ChatCollection Class Reference

This class manages *Chat* collections. More...

```sql
import "Skype4COM.idl";
```
Detailed Description

This class manages Chat collections.

This class manages collections of Chat objects. To query a chat collection, use For Each .. Next statements.

Example:

```csharp
'// To query chat history:
For Each oChat In oSkype.Chats
Next
```

See also:

IChat, IChatCollection

The documentation for this class was generated from the following file:

- Skype4COM.idl

Copyright © 2006 Skype Limited. All rights reserved.
# Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**SKYPE4COMLib**

**ChatMember**

---

**Overview:**

- **SKYPE4COMLib:** A library for integrating Skype 4 COM into applications.
- **ChatMember:** A class within the SKYPE4COMLib for managing chat members in Skype.

**Features:**

- **Communications:** Supports communication within chat environments.
- **Integration:** Facilitates seamless integration of Skype functionalities into custom software applications.

**Usage:**

- Ideal for developers looking to enhance their applications with Skype integration capabilities.
- Provides a robust framework for managing chat members and their interactions.

---

**Technical Details:**

- **Version:** 1.0.36.0
- **Compatibility:** Across various operating systems supporting Skype.
- **Documentation:** Comprehensive API documentation and examples available for developers.
ChatMember Class Reference

This class manages chat member. More...

import "Skype4COM.idl";
Detailed Description

This class manages chat member.

See also:

IChatMember

The documentation for this class was generated from the following file:

- Skype4COM.idl

Copyright © 2006 Skype Limited. All rights reserved.
## Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**SKYPE4COMLib**  **ChatMemberCollection**
ChatMemberCollection Class Reference

This class manages chat member collections. More...

import "Skype4COM.idl";
Detailed Description

This class manages chat member collections.

See also:

IChatMemberCollection

The documentation for this class was generated from the following file:

- Skype4COM.idl

Copyright © 2006 Skype Limited. All rights reserved.
# Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Files</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class List</td>
<td>Class Members</td>
</tr>
</tbody>
</table>

**SKYPE4COMLib**  ChatMessage

**SKYPE4COMLib** is a library that provides a COM interface to Skype.
This class manages chat messages. More...

import "Skype4COM.idl";
Detailed Description

This class manages chat messages.

See also:
  IChatMessage

The documentation for this class was generated from the following file:

- Skype4COM.idl

Copyright © 2006 Skype Limited. All rights reserved.
<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

SKYPE4COMLib  ChatMessageCollection
ChatMessageCollection Class Reference

This class manages chat message collections. More...

import "Skype4COM.idl";
Detailed Description

This class manages chat message collections.

See also:

IChatMessageCollection

The documentation for this class was generated from the following file:

- Skype4COM.idl

Copyright © 2006 Skype Limited. All rights reserved.
<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SKYPE4COMLib</td>
<td>Client</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Client Class Reference

This class controls features related to the Skype user interface. More...

import "Skype4COM.idl";
Detailed Description

This class controls features related to the Skype user interface.

See also:
   IClient

The documentation for this class was generated from the following file:

- Skype4COM.idl

Copyright © 2006 Skype Limited. All rights reserved.
## Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**SKYPE4COMLib** Command
Command Class Reference

This class manages Skype API command objects. More...

import "Skype4COM.idl";
Detailed Description

This class manages Skype API command objects.

See also:
   ICommand

The documentation for this class was generated from the following file:

- Skype4COM.idl

Copyright © 2006 Skype Limited. All rights reserved.
<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SKYPE4COMLib</td>
<td>Conference</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Conference Class Reference

This class manages conference calls. More...

import "Skype4COM.idl";
**Detailed Description**

This class manages conference calls.

The *Conference* class manages a call connected to multiple participants.

**See also:**

*IConference, IConferenceCollection*

The documentation for this class was generated from the following file:

- *Skype4COM.idl*

---

Copyright © 2006 *Skype Limited*. All rights reserved.
## Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**SKYPE4COMLib** > **ConferenceCollection**
This class manages conference collections. More...

import "Skype4COM.idl";
Detailed Description

This class manages conference collections.

Use the ConferenceCollection class to query conference collections in call history. To query a collection class, use For Each .. Next statements.

See also:
   IConference, IConferenceCollection, ISkype::Conferences

The documentation for this class was generated from the following file:

- Skype4COM.idl

Copyright © 2006 Skype Limited. All rights reserved.
# Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SKYPE4COMLib Conversion</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Conversion Class Reference

This class converts text to constant values and converts constant values to text. More...

import "Skype4COM.idl";
**Detailed Description**

This class converts text to constant values and converts constant values to text.

**Example:**

```vbscript
'// To create an object:
Set oSkype = CreateObject("Skype4COM.Skype")

'// To convert an attachment status to text:
oAttachStatus = oSkype.Convert.AttachmentStatusToText(oSkype.AttachmentStatus)

'// To change the online status of a user:
```

**See also:**

IConversion, ISkype::Convert

**Deprecated:**

The `SKYPE4COMLib::Conversion` class is deprecated, support for multilanguage status codes support will be dropped in the future.

The documentation for this class was generated from the following file:

- Skype4COM.idl

---

Copyright ♦ 2006 Skype Limited. All rights reserved.
Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SKYPE4COMLib</td>
<td>Group</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Group Class Reference

This class manages contact groups. More...

import "Skype4COM.idl";
Detailed Description

This class manages contact groups.

See also:

IGroup

The documentation for this class was generated from the following file:

- Skype4COM.idl

Copyright © 2006 Skype Limited. All rights reserved.
# Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SKYPE4COMLib</td>
<td>GroupCollection</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
GroupCollection Class Reference

This class manages collections of Group objects. More...

import "Skype4COM.idl";
Detailed Description

This class manages collections of Group objects.

See also:
IGroupCollection, IGroup

The documentation for this class was generated from the following file:

- Skype4COM.idl

Copyright © 2006 Skype Limited. All rights reserved.
## Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Public Member Functions | Properties
IApplication Interface Reference

**Application** class default interface. More...

```
import "Skype4COM.idl"
```

List of all members.
## Public Member Functions

<table>
<thead>
<tr>
<th>HRESULT</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HRESULT</td>
<td><strong>Create ()</strong></td>
<td>This command creates the application context.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>Delete ()</strong></td>
<td>This command deletes the application context.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>Connect ([in] BSTR Username,[in, defaultvalue(0)] VARIANT_BOOL WaitConnected)</strong></td>
<td>This command connects a user to the application context.</td>
</tr>
<tr>
<td>HRESULT</td>
<td>*<em>SendDatagram ([in] BSTR Text,[in, defaultvalue(0)] IApplicationStreamCollection <em>pStreams)</em></em></td>
<td>This command broadcasts a datagram to connected streams.</td>
</tr>
</tbody>
</table>
## Properties

<table>
<thead>
<tr>
<th>Type</th>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BSTR</td>
<td>Name [get]</td>
<td>This command queries an application name.</td>
</tr>
<tr>
<td>IApplicationStreamCollection</td>
<td>Streams [get]</td>
<td>This command queries application streams.</td>
</tr>
<tr>
<td>IUserCollection</td>
<td>ConnectableUsers [get]</td>
<td>This command queries users that can connect to the application.</td>
</tr>
<tr>
<td>IUserCollection</td>
<td>ConnectingUsers [get]</td>
<td>This command queries the users who are currently connecting to the application.</td>
</tr>
<tr>
<td>IApplicationStreamCollection</td>
<td>SendingStreams [get]</td>
<td>This command queries application sending streams.</td>
</tr>
<tr>
<td>IApplicationStreamCollection</td>
<td>ReceivedStreams [get]</td>
<td>This command queries application receiving streams.</td>
</tr>
</tbody>
</table>
Detailed Description

**Application** class default interface.

See also:

*IApplicationStream*
Member Function Documentation

HRESULT Connect ([in] BSTR Username, [in, defaultvalue(0)] VARIANT_BOOL WaitConnected)

This command connects a user to the application context.

Parameters:
- [in] Username contains the Skypename of the user to connect.
- [in] WaitConnected is an optional Boolean value to indicate whether to wait for the user to connect or not.

HRESULT SendDatagram ([in] BSTR Text, [in, defaultvalue(0)] IApplicationStreamCollection pStreams)

This command broadcasts a datagram to connected streams.

Parameters:
- [in] pStreams is an optional parameter you can use to specify the collection of streams that will receive the datagram.

The documentation for this interface was generated from the following file:

- Skype4COM.idl

Copyright ✉ 2006 Skype Limited. All rights reserved.
Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td>Class List</td>
<td>Class Members</td>
</tr>
</tbody>
</table>

Public Member Functions | Properties
ApplicationStream class default interface. More...

import "Skype4COM.idl";

List of all members.
## Public Member Functions

<table>
<thead>
<tr>
<th>HRESULT</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HRESULT</td>
<td><strong>Read</strong> ([out, retval] BSTR *pText)</td>
<td>This command reads a text string from an application stream.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>Write</strong> ([in] BSTR Text)</td>
<td>This command writes a text string to an application stream.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>SendDatagram</strong> ([in] BSTR Text)</td>
<td>This command sends a datagram message. Delivery of the message is not guaranteed.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>Disconnect</strong> ()</td>
<td>This command disconnects an application stream from an application context.</td>
</tr>
</tbody>
</table>
### Properties

<table>
<thead>
<tr>
<th>Type</th>
<th>Property</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BSTR</td>
<td>ApplicationName</td>
<td>[get]</td>
<td>This command queries an application name.</td>
</tr>
<tr>
<td>BSTR</td>
<td>Handle</td>
<td>[get]</td>
<td>This command queries the name of an application stream.</td>
</tr>
<tr>
<td>LONG</td>
<td>DataLength</td>
<td>[get]</td>
<td>This command queries the data length of an application stream.</td>
</tr>
<tr>
<td>BSTR</td>
<td>PartnerHandle</td>
<td>[get]</td>
<td>This command queries the Skypename of the sender of an application stream.</td>
</tr>
</tbody>
</table>
Detailed Description

**ApplicationStream** class default interface.

See also:

*IApplication*
Member Function Documentation

HRESULT Read ([out, retval] BSTR * pText)

This command reads a text string from an application stream.

Parameters:
   [out] pText receives the text string from the application stream.

HRESULT SendDatagram ([in] BSTR Text)

This command sends a datagram message. Delivery of the message is not guaranteed.

Parameters:
   [in] Text contains the message to send.

Note:
   To transmit binary data use the base64 encoding to convert your data to strings before passing those strings to IApplication.SendDatagram or IApplicationStream.Write methods.

Example:

```javascript
//Declare the variables
var Skype, App;

//Create the Skype application object.
Skype = new ActiveXObject("Skype4COM.Skype");

// Connect events:
WScript.ConnectObject(Skype, "Skype_");
```
//Create the application object
var App = Skype.Application("TestBase64DataTransfer");

//Create the application context
App.Create();
WScript.Echo("App created.");

//Connect remote user
if (WScript.Arguments.Count() > 0)
{
    App.Connect(WScript.Arguments.Item(0), true);
    WScript.Echo(WScript.Arguments.Item(0) + " connected.");
}

//Byte array
var byteArray = new Array(17, 34, 51, 68, 85, 102, 119, 136, 153, 170);

// Send datagram
if (App.Streams.Count)
{
    //Encoded byte array
    var encoded = encode64(byteArray);

    //Send encoded array to streams
    for (var i=0; i < App.Streams.Count; i++)
    {
        var stm = App.Streams.Item(i+1);
        for (var i=0; i < App.Streams.Count; i++)
        {
            stm.SendDatagram(encoded);
        }
    }

    WScript.Echo(">Datagram [" + byteArray + "] sent");
}
}  
//Sleep 10 minutes  
WScript.Sleep(600000);

//Application datagram events  
function Skype_ApplicationDatagram(aApp, aStream, aText)  
{  
    WScript.Echo(">Datagram [" + decode64(aText) + "] received");  
}

//Here's the encode function  
function encode64(inp)  
{  
    // Read more about base64 at http://en.wikipedia.org/wiki/Base64  
    ...  
}

//Here's the decode function  
function decode64(inp)  
{  
    ...  
}

HRESULT Write ([in] BSTR Text)  

This command writes a text string to an application stream.  

Parameters:  
file:

- Skype4COM.idl

Copyright © 2006 Skype Limited. All rights reserved.
Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Public Member Functions | Properties
IApplicationStreamCollection Interface Reference

ApplicationStreamCollection class default interface. More...

import "Skype4COM.idl";

List of all members.
## Public Member Functions

<table>
<thead>
<tr>
<th>HRESULT</th>
<th><strong>Add</strong> ([in] IApplicationStream *plItem)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>This command adds a new item to an ApplicationStreamCollection.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>HRESULT</th>
<th><strong>Remove</strong> ([in] long Index)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>This command removes a stream from an ApplicationStreamCollection.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>HRESULT</th>
<th><strong>RemoveAll</strong> (void)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>This command removes all streams from an ApplicationStreamCollection.</td>
</tr>
</tbody>
</table>
### Properties

<table>
<thead>
<tr>
<th>Type</th>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LONG</td>
<td><strong>Count</strong> [get]</td>
<td>This command queries the number of streams in an ApplicationStreamCollection.</td>
</tr>
<tr>
<td>IApplicationStream</td>
<td><strong>Item</strong> ([in] LONG Index) [get]</td>
<td>This command queries a stream with a specified index in an ApplicationStreamCollection.</td>
</tr>
<tr>
<td>IUnknown</td>
<td><strong>_NewEnum</strong> [get]</td>
<td>This command queries an ApplicationStreamCollection enumerator object.</td>
</tr>
</tbody>
</table>
Detailed Description

**ApplicationStreamCollection** class default interface.

**See also:**
- `IApplicationStream`
Member Function Documentation

HRESULT Add ([in] IApplicationStream * pStream )

This command adds a new item to an ApplicationStreamCollection.

Parameters:
   [in] pStream contains the ApplicationStream to add.

HRESULT Remove ([in] long Index )

This command removes a stream from an ApplicationStreamCollection.

Parameters:
   [in] Index contains the item to be removed.
Property Documentation

HRESULT Item [get]

This command queries a stream with a specified index in an ApplicationStreamCollection.

Parameters:

[in] Index contains the stream to be returned.

The documentation for this interface was generated from the following file:

- Skype4COM.idl

Copyright © 2006 Skype Limited. All rights reserved.
# Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Files</strong></td>
<td><strong>Examples</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Public Member Functions
Properties
ICall Interface Reference

**Call** class default interface. More...

import "Skype4COM.idl";

List of all members.
## Public Member Functions

<table>
<thead>
<tr>
<th>HRESULT</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HRESULT</td>
<td><strong>Hold ()</strong></td>
<td>This command places a call on hold.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>Resume ()</strong></td>
<td>This command resumes a call that was on hold.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>Finish ()</strong></td>
<td>This command ends a call.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>Answer ()</strong></td>
<td>This command answers a call.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>Join ([in] LONG Id)</strong></td>
<td>This command joins a call with another call to form a call conference.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>StartVideoSend ()</strong></td>
<td>This command starts video send.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>StopVideoSend ()</strong></td>
<td>This command stops video send.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>StartVideoReceive ()</strong></td>
<td>This command starts video receive.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>StopVideoReceive ()</strong></td>
<td>This command stops video receive.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>RedirectToVoicemail ()</strong></td>
<td>This command redirects an incoming call to voicemail.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>Forward ()</strong></td>
<td>This command forwards an incoming call.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>Transfer ([in] BSTR Target)</strong></td>
<td>Transfers call to contact or phone number.</td>
</tr>
</tbody>
</table>
### Properties

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Property Name</th>
<th>Access Mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>LONG</code></td>
<td><strong>Id</strong> [get]</td>
<td></td>
<td>This command queries the call ID.</td>
</tr>
<tr>
<td><code>DATE</code></td>
<td><strong>Timestamp</strong> [get]</td>
<td></td>
<td>This command queries the call timestamp.</td>
</tr>
<tr>
<td><code>BSTR</code></td>
<td><strong>PartnerHandle</strong> [get]</td>
<td></td>
<td>This command queries the Skypename of a remote caller.</td>
</tr>
<tr>
<td><code>BSTR</code></td>
<td><strong>PartnerDisplayName</strong> [get]</td>
<td></td>
<td>This command queries the name of the remote caller.</td>
</tr>
<tr>
<td><code>LONG</code></td>
<td><strong>ConferenceId</strong> [get]</td>
<td></td>
<td>This command queries a call conference ID.</td>
</tr>
<tr>
<td><code>TCallType</code></td>
<td><strong>Type</strong> [get]</td>
<td></td>
<td>This command queries a call type.</td>
</tr>
<tr>
<td><code>TCallStatus</code></td>
<td><strong>Status</strong> [get, set]</td>
<td></td>
<td>This command queries/sets a call status.</td>
</tr>
<tr>
<td><code>TCallFailureReason</code></td>
<td><strong>FailureReason</strong> [get]</td>
<td></td>
<td>This command queries a call failure reason.</td>
</tr>
<tr>
<td><code>BSTR</code></td>
<td><strong>Subject</strong> [get]</td>
<td></td>
<td>This command queries the subject of a call.</td>
</tr>
<tr>
<td><code>BSTR</code></td>
<td><strong>PstnNumber</strong> [get]</td>
<td></td>
<td>This command queries the PSTN number of a call.</td>
</tr>
<tr>
<td><code>LONG</code></td>
<td><strong>Duration</strong> [get]</td>
<td></td>
<td>This command queries the duration of a call.</td>
</tr>
<tr>
<td><code>BSTR</code></td>
<td><strong>PstnStatus</strong> [get]</td>
<td></td>
<td>This command queries the call PSTN status.</td>
</tr>
<tr>
<td><code>VARIANT_BOOL</code></td>
<td><strong>Seen</strong> [get, set]</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Property</strong></td>
<td><strong>Type</strong></td>
<td><strong>Description</strong></td>
<td></td>
</tr>
<tr>
<td>--------------</td>
<td>----------</td>
<td>-----------------</td>
<td></td>
</tr>
<tr>
<td><strong>seen</strong></td>
<td><strong>BSTR</strong></td>
<td>This command queries or sets the call to a status of seen.</td>
<td></td>
</tr>
<tr>
<td><strong>DTMF</strong></td>
<td><strong>BSTR</strong></td>
<td>Set/Get, This command sends a DTMF tone.</td>
<td></td>
</tr>
<tr>
<td><strong>Participants</strong></td>
<td><strong>IParticipantCollection</strong></td>
<td>Get, This command queries the number of participants in a conference call not hosted by the user.</td>
<td></td>
</tr>
<tr>
<td><strong>VmDuration</strong></td>
<td><strong>LONG</strong></td>
<td>Get, This command returns the duration of a voicemail.</td>
<td></td>
</tr>
<tr>
<td><strong>VmAllowedDuration</strong></td>
<td><strong>LONG</strong></td>
<td>Get, This command returns the permitted duration of a voicemail.</td>
<td></td>
</tr>
<tr>
<td><strong>VideoStatus</strong></td>
<td><strong>TCallVideoStatus</strong></td>
<td>Get, This command queries call video status.</td>
<td></td>
</tr>
<tr>
<td><strong>VideoSendStatus</strong></td>
<td><strong>TCallVideoSendStatus</strong></td>
<td>Get, This command queries call video send status.</td>
<td></td>
</tr>
<tr>
<td><strong>VideoReceiveStatus</strong></td>
<td><strong>TCallVideoSendStatus</strong></td>
<td>Get, This command queries call video receive status.</td>
<td></td>
</tr>
<tr>
<td><strong>Rate</strong></td>
<td><strong>LONG</strong></td>
<td>Get, This command queries the call rate, expressed in cents.</td>
<td></td>
</tr>
<tr>
<td><strong>RateCurrency</strong></td>
<td><strong>BSTR</strong></td>
<td>Get, This command queries currency code for the call rate.</td>
<td></td>
</tr>
<tr>
<td><strong>RatePrecision</strong></td>
<td><strong>LONG</strong></td>
<td>Get, This command queries the call rate precision.</td>
<td></td>
</tr>
<tr>
<td><strong>InputDevice</strong></td>
<td><strong>BSTR</strong></td>
<td>Get/Set, This command queries or sets the input sound device.</td>
<td></td>
</tr>
<tr>
<td>Data Type</td>
<td>Method Name</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>------------</td>
<td>-----------------------------------</td>
<td>------------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>BSTR</td>
<td><strong>OutputDevice</strong> ([in] TCallIoDeviceType DeviceType) [get, set]</td>
<td>This command queries or sets the output sound device.</td>
<td></td>
</tr>
<tr>
<td>BSTR</td>
<td><strong>CaptureMicDevice</strong> ([in] TCallIoDeviceType DeviceType) [get, set]</td>
<td>This command queries or sets the mic capture device.</td>
<td></td>
</tr>
<tr>
<td>VARIANT_BOOL</td>
<td><strong>InputStatus</strong> [get]</td>
<td>This command queries the call voice input status.</td>
<td></td>
</tr>
<tr>
<td>BSTR</td>
<td><strong>ForwardedBy</strong> [get]</td>
<td>This command queries the identity of a user who forwards a call.</td>
<td></td>
</tr>
<tr>
<td>VARIANT_BOOL</td>
<td><strong>CanTransfer</strong> ([in] BSTR Target) [get]</td>
<td>This command queries if a call can be transferred to contact or number.</td>
<td></td>
</tr>
<tr>
<td>TCallStatus</td>
<td><strong>TransferStatus</strong> [get]</td>
<td>Returns call transferring status.</td>
<td></td>
</tr>
<tr>
<td>VARIANT_BOOL</td>
<td><strong>TransferActive</strong> [get]</td>
<td>Returns true if the call has been transferred.</td>
<td></td>
</tr>
<tr>
<td>BSTR</td>
<td><strong>TransferredBy</strong> [get]</td>
<td>Returns the Skypename of the user who transferred the call.</td>
<td></td>
</tr>
<tr>
<td>BSTR</td>
<td><strong>TransferredTo</strong> [get]</td>
<td>Returns the Skypename of the user or phone number the call has been transferred to.</td>
<td></td>
</tr>
<tr>
<td>BSTR</td>
<td><strong>TargetIdentity</strong> [get]</td>
<td>Returns the target number for incoming SkypeIN calls.</td>
<td></td>
</tr>
</tbody>
</table>
Detailed Description

**Call** class default interface.
Member Function Documentation

HRESULT Join ([in] LONG Id )

This command joins a call with another call to form a call conference.

Parameters:
[in] Id contains the ID of the other call to join to the conference.

HRESULT Transfer ([in] BSTR Target )

Transfers call to contact or phone number.

Parameters:
[in] Target phone number(s) or Skypename(s) the call is being transferred to.

Note:
You can transfer an incoming call to a group by specifying a comma separated list of targets, first one of the group to answer will get the call.

See also:
ICall::CanTransfer
Property Documentation

HRESULT CanTransfer [get]

This command queries if a call can be transferred to contact or number.

Parameters:

[in] Target Skypename or phone number the call to be transferre to.

See also:
ICall::Transfer

HRESULT CaptureMicDevice [get, set]

This command queries or sets the mic capture device.

Parameters:

[in] DeviceType contains the device type, TCallIoDeviceType::callIoDeviceTypeSound cannot be used with this property.

Note:
This command functions for active calls only.

HRESULT DTMF [set]

This command sends a DTMF tone.

Parameters:

[in] newVal contains the DTMF value to send. Permitted symbols are: {0..9,#,*}.

Note:
This command functions for active calls only.

**HRESULT InputDevice [get, set]**

This command queries or sets the input sound device.

**Parameters:**
- *in* `DeviceType` contains the device type.

**Note:**
- This command functions for active calls only.

**HRESULT OutputDevice [get, set]**

This command queries or sets the output sound device.

**Parameters:**
- *in* `DeviceType` contains the device type.

**Note:**
- This command functions for active calls only.

**HRESULT Seen [get, set]**

This command queries or sets the call to a status of seen.

**Parameters:**
- *in* `newVal` contains a Boolean value to indicate if a call is seen or not.

**Note:**
- You cannot alter the call seen status from seen to unseen.

**HRESULT Status [get, set]**
This command queries/sets a call status.

**Deprecated:**
Set `ICall::Status` is deprecated, use the `ICall::Hold`, `ICall::Resume`, `ICall::Finish` or `ICall::Answer` instead.

The documentation for this interface was generated from the following file:

- Skype4COM.idl

---

Copyright  ℹ️ 2006 Skype Limited. All rights reserved.
Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Public Member Functions | Properties
ICallChannel Interface Reference

**CallChannel** class default interface. More...

import "Skype4COM.idl";

List of all members.
Public Member Functions

<table>
<thead>
<tr>
<th>HRESULT</th>
<th><strong>SendTextMessage</strong> (BSTR Text)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>This command sends a <strong>CallChannel</strong> message.</td>
</tr>
</tbody>
</table>
### Properties

<table>
<thead>
<tr>
<th><strong>TCallChannelType</strong></th>
<th><strong>Type</strong> [get, set]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>This command queries/sets the <strong>CallChannel</strong> type.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>IApplicationStream</strong></th>
<th><strong>Stream</strong> [get]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>This command returns the channel <strong>ApplicationStream</strong> object.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>ICallChannelManager</strong></th>
<th><strong>Manager</strong> [get]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>This command returns the related <strong>CallChannelManager</strong> object.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>ICall</strong></th>
<th><strong>Call</strong> [get]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>This command returns the related <strong>Call</strong> object.</td>
</tr>
</tbody>
</table>
Detailed Description

CallChannel class default interface.
Member Function Documentation

HRESULT SendTextMessage ( BSTR Text )

This command sends a CallChannel message.

Parameters:

The documentation for this interface was generated from the following file:

- Skype4COM.idl

Copyright © 2006 Skype Limited. All rights reserved.
Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td>Class List</td>
<td>Class Members</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Public Member Functions</td>
<td>Properties</td>
</tr>
</tbody>
</table>
ICallChannelCollection Interface Reference

CallChannelCollection class default interface. More...

import "Skype4COM.idl";

List of all members.
**Public Member Functions**

<table>
<thead>
<tr>
<th>HRESULT</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HRESULT</td>
<td><strong>Add</strong> ([in] ICallChannel *pChannel)</td>
<td>This command adds a new call channel to a CallChannelCollection.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>Remove</strong> ([in] long Index)</td>
<td>This command removes an item from a CallChannelCollection.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>RemoveAll</strong> (void)</td>
<td>This command removes all items from a CallChannelCollection.</td>
</tr>
</tbody>
</table>
### Properties

<table>
<thead>
<tr>
<th>Type</th>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LONG</td>
<td><strong>Count</strong> [get]</td>
<td>This command queries the number of call channels in a <code>CallChannelCollection</code>.</td>
</tr>
<tr>
<td><strong>ICallChannel</strong></td>
<td><strong>Item ([in] LONG Index) [get]</strong></td>
<td>This command queries a <code>CallChannelCollection</code> item with a specified index.</td>
</tr>
<tr>
<td><strong>IUnknown</strong></td>
<td><strong>_NewEnum</strong> [get]</td>
<td>This command queries a <code>CallChannelCollection</code> enumerator object.</td>
</tr>
</tbody>
</table>
Detailed Description

`CallChannelCollection` class default interface.
**Member Function Documentation**

**HRESULT Add ([in] ICallChannel * pItem)**

This command adds a new call channel to a `CallChannelCollection`.

**Parameters:**
- **[in]** `pItem` contains the interface pointer for the `CallChannel` object.

**HRESULT Remove ([in] long Index)**

This command removes an item from a `CallChannelCollection`.

**Parameters:**
- **[in]** `Index` contains the item to be removed.
Property Documentation

HRESULT Item [get]

This command queries a CallChannelCollection item with a specified index.

Parameters:

[in] Index contains the item to be returned.

The documentation for this interface was generated from the following file:

- Skype4COM.idl

Copyright 2006 Skype Limited. All rights reserved.
<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Public Member Functions | Properties
ICallChannelManager Interface Reference

**CallChannelManager** class default interface. More...

import "Skype4COM.idl";

List of all members.
### Public Member Functions

<table>
<thead>
<tr>
<th>HRESULT</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><strong>Connect</strong> ([in] ISkype *pSkype)</td>
<td>This command binds the <strong>CallChannelManager</strong> object to the <strong>Skype</strong> object.</td>
</tr>
<tr>
<td></td>
<td><strong>Disconnect</strong> ()</td>
<td>This command disconnects an existing binding between the <strong>CallChannelManager</strong> and the <strong>Skype</strong> object.</td>
</tr>
<tr>
<td></td>
<td><strong>CreateApplication</strong> ([in, defaultvalue(&quot;&quot;)] BSTR ApplicationName)</td>
<td>This command creates the <strong>Application</strong> context.</td>
</tr>
</tbody>
</table>
### Properties

<table>
<thead>
<tr>
<th>Interface</th>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICallChannelCollection</td>
<td>Channels</td>
<td>[get] This command queries call data channels.</td>
</tr>
<tr>
<td>TCallChannelType</td>
<td>ChannelType</td>
<td>[get, set] This command queries/sets the default call channel type.</td>
</tr>
<tr>
<td>BSTR</td>
<td>Name</td>
<td>[get, set] This command queries/sets the Application context name.</td>
</tr>
<tr>
<td>VARIANT_BOOL</td>
<td>Created</td>
<td>[get] This command returns the application context creation status.</td>
</tr>
</tbody>
</table>
Detailed Description

**CallChannelManager** class default interface.
Member Function Documentation

HRESULT Connect ([in] ISkype * pSkype )

This command binds the CallChannelManager object to the Skype object.

Parameters:
   [in] pSkype contains the interface pointer for the Skype object.

HRESULT CreateApplication ([in, defaultvalue(""')] BSTR ApplicationName)

This command creates the Application context.

Parameters:
   [in] ApplicationName contains an optional application context name.

The documentation for this interface was generated from the following file:

- Skype4COM.idl

Copyright © 2006 Skype Limited. All rights reserved.
Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Properties
ICallChannelMessage Interface Reference

CallChannelMessage class default interface. More...

import "Skype4COM.idl";

List of all members.
## Properties

<table>
<thead>
<tr>
<th>BSTR</th>
<th><strong>Text</strong> [get, set]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>This command queries/sets message text.</td>
</tr>
</tbody>
</table>
Detailed Description

**CallChannelMessage** class default interface.

The documentation for this interface was generated from the following file:

- **Skype4COM.idl**

Copyright © 2006 Skype Limited. All rights reserved.
<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Public Member Functions | Properties
ICallCollection Interface Reference

**ICallCollection** class default interface.  More...

import "Skype4COM.idl";

List of all members.
Public Member Functions

<table>
<thead>
<tr>
<th>HRESULT</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HRESULT Add ([in] ICall *pCall)</td>
<td>This command adds a new call to a CallCollection.</td>
<td></td>
</tr>
<tr>
<td>HRESULT Remove ([in] long Index)</td>
<td>This command removes a call from a CallCollection.</td>
<td></td>
</tr>
<tr>
<td>HRESULT RemoveAll (void)</td>
<td>This command removes all calls from a CallCollection.</td>
<td></td>
</tr>
</tbody>
</table>
## Properties

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LONG</td>
<td>Count [get]</td>
<td>This command queries the number of calls in a CallCollection.</td>
</tr>
<tr>
<td>ICall</td>
<td>Item ([in] LONG Index) [get]</td>
<td>This command queries a call with a specified index in a CallCollection.</td>
</tr>
<tr>
<td>IUnknown</td>
<td>_NewEnum [get]</td>
<td>This command queries the enumerator object of a CallCollection.</td>
</tr>
</tbody>
</table>
Detailed Description

**CallCollection** class default interface.
**Member Function Documentation**

**HRESULT Add ([in] ICall * pCall )**

This command adds a new call to a **CallCollection**.

**Parameters:**
- **[in] pCall** receives the interface pointer for the **Call** object.

**HRESULT Remove ([in] long Index )**

This command removes a call from a **CallCollection**.

**Parameters:**
- **[in] Index** contains the call to be removed.
Property Documentation

HRESULT Item [get]

This command queries a call with a specified index in a CallCollection.

Parameters:
    [in] Index contains the call to be returned.

The documentation for this interface was generated from the following file:

- Skype4COM.idl

Copyright 2006 Skype Limited. All rights reserved.
## Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Public Member Functions
Properties
IChat Interface Reference

Chat class default interface. More...

import "Skype4COM.idl";

List of all members.
### Public Member Functions

<table>
<thead>
<tr>
<th>HRESULT</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HRESULT</td>
<td><code>OpenWindow()</code></td>
<td>This command opens a chat window.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><code>SendMessage</code> ([in] BSTR MessageText,[out, retval] IChatMessage **pVal)`</td>
<td>This command sends a chat message.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><code>Leave()</code></td>
<td>This command leaves a chat.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><code>AddMembers</code> ([in] IUserCollection *pMembers)`</td>
<td>This command adds new members to a chat.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><code>Bookmark()</code></td>
<td>This command bookmarks a chat.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><code>Unbookmark()</code></td>
<td>This command removes the bookmark for a chat.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><code>SetPassword</code> ([in] BSTR Password,[in, defaultvalue(&quot;&quot;)] BSTR Hint)`</td>
<td>Sets chat password.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><code>Join</code> (void)</td>
<td>Joins the chat.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><code>Kick</code> ([in] BSTR Handle)</td>
<td>Kicks member from chat.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><code>KickBan</code> ([in] BSTR Handle)</td>
<td>Kicks and ban member from chat.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><code>Disband</code> (void)</td>
<td>Ends chat.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><code>EnterPassword</code> ([in] BSTR Password)`</td>
<td>Enters chat password.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><code>ClearRecentMessages</code> (void)</td>
<td>Clears recent chat messages.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><code>AcceptAdd</code> (void)</td>
<td>Accepts shared group add.</td>
</tr>
</tbody>
</table>
### Properties

<table>
<thead>
<tr>
<th>Interface/Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>BSTR</strong> Name [get]</td>
<td>This command queries the chat name.</td>
</tr>
<tr>
<td><strong>IChatMessageCollection</strong> Messages [get]</td>
<td>This command queries chat messages.</td>
</tr>
<tr>
<td><strong>DATE</strong> Timestamp [get]</td>
<td>This command queries the chat timestamp.</td>
</tr>
<tr>
<td><strong>IUser</strong> Adder [get]</td>
<td>This command queries who added a user to a chat.</td>
</tr>
<tr>
<td><strong>TChatStatus</strong> Status [get]</td>
<td>This command queries a chat status.</td>
</tr>
<tr>
<td><strong>IUserCollection</strong> Posters [get]</td>
<td>This command queries users who have posted messages to a chat.</td>
</tr>
<tr>
<td><strong>IUserCollection</strong> Members [get]</td>
<td>This command queries the members of a chat.</td>
</tr>
<tr>
<td><strong>BSTR</strong> Topic [get, set]</td>
<td>This command queries/sets a chat topic.</td>
</tr>
<tr>
<td><strong>IUserCollection</strong> ActiveMembers [get]</td>
<td>This command queries the active members of a chat.</td>
</tr>
<tr>
<td><strong>BSTR</strong> FriendlyName [get]</td>
<td>This command queries the &quot;friendly&quot; name of a chat.</td>
</tr>
<tr>
<td><strong>IChatMessageCollection</strong> RecentMessages [get]</td>
<td>This command queries the most recent chat messages.</td>
</tr>
<tr>
<td><strong>VARIANT_BOOL</strong> Bookmarked [get]</td>
<td>This command queries if a chat is bookmarked.</td>
</tr>
<tr>
<td>Class</td>
<td>Property</td>
</tr>
<tr>
<td>----------------------------</td>
<td>-------------------------------</td>
</tr>
<tr>
<td>BSTR</td>
<td><strong>TopicXML</strong> [get, set]</td>
</tr>
<tr>
<td>IChatMemberCollection</td>
<td><strong>MemberObjects</strong> [get]</td>
</tr>
<tr>
<td>BSTR</td>
<td><strong>Blob</strong> [get]</td>
</tr>
<tr>
<td>LONG</td>
<td><strong>Options</strong> [get, set]</td>
</tr>
<tr>
<td>BSTR</td>
<td><strong>PasswordHint</strong> [get]</td>
</tr>
<tr>
<td>BSTR</td>
<td><strong>Description</strong> [get, set]</td>
</tr>
<tr>
<td>BSTR</td>
<td><strong>DialogPartner</strong> [get]</td>
</tr>
<tr>
<td>DATE</td>
<td><strong>ActivityTimestamp</strong> [get]</td>
</tr>
<tr>
<td>TChatMemberRole</td>
<td><strong>MyRole</strong> [get]</td>
</tr>
<tr>
<td>IUserCollection</td>
<td><strong>Applicants</strong> [get]</td>
</tr>
<tr>
<td>BSTR</td>
<td><strong>AlertString</strong> [set]</td>
</tr>
<tr>
<td>TChatType</td>
<td><strong>Type</strong> [get]</td>
</tr>
<tr>
<td>TChatMyStatus</td>
<td><strong>MyStatus</strong> [get]</td>
</tr>
</tbody>
</table>
Detailed Description

Chat class default interface.
Member Function Documentation

**HRESULT AddMembers ([in] IUserCollection * pMembers)**

This command adds new members to a chat.

**Parameters:**
- **[in] pMembers** receives the interface pointer for the `UserCollection` object.

**HRESULT EnterPassword ([in] BSTR Password)**

Enters chat password.

**Parameters:**
- **[in] Password** contains password string.

**HRESULT SendMessage ([in] BSTR MessageText, [out, retval] IChatMessage ** pVal)**

This command sends a chat message.

**Parameters:**
- **[in] MessageText** contains the message text.
- **[out] pVal** receives the interface pointer for the `ChatMessage` object.

**HRESULT SetPassword ([in] BSTR Password, [in, defaultvalue(""')] BSTR Hint)**
Sets chat password.

**Parameters:**
- **[in]** **Password** contains password.
- **[in]** **Hint** contains password hint.

**See also:**
- IChat::PasswordHint
### Property Documentation

#### HRESULT Options [get, set]

Returns/sets chat options.

**See also:**
- TChatOptions

#### HRESULT Status [get]

This command queries a chat status.

**Deprecated:**
- ICall::Status is deprecated, use the IChat::Type and IChat::MyStatus instead.

The documentation for this interface was generated from the following file:

- **Skype4COM.idl**

---

Copyright © 2006 Skype Limited. All rights reserved.
<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Class List</td>
<td>Class Members</td>
</tr>
</tbody>
</table>

Public Member Functions | Properties
ChatCollection class default interface. More...

import "Skype4COM.idl";

List of all members.
## Public Member Functions

<table>
<thead>
<tr>
<th>HRESULT</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add ([in] IChat *pItem)</td>
<td>HRESULT</td>
<td>This command adds a new chat to a ChatCollection.</td>
</tr>
<tr>
<td>Remove ([in] long Index)</td>
<td>HRESULT</td>
<td>This command removes an item from a ChatCollection.</td>
</tr>
<tr>
<td>RemoveAll (void)</td>
<td>HRESULT</td>
<td>This command removes all items from a ChatCollection.</td>
</tr>
</tbody>
</table>
### Properties

<table>
<thead>
<tr>
<th>LONG</th>
<th><strong>Count</strong> [get]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>This command queries the number of chats in a <em>ChatCollection</em>.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IChat</th>
<th><strong>Item ([in] LONG Index) [get]</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>This command queries a chat collection item with a specified index.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IUnknown</th>
<th><strong>NewEnum [get]</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>This command queries a <em>ChatCollection</em> enumerator object.</td>
</tr>
</tbody>
</table>
Detailed Description

ChatCollection class default interface.
**Member Function Documentation**

**HRESULT Add ([in] IChat * pItem )**

This command adds a new chat to a `ChatCollection`.

**Parameters:**
- **[in] pItem** contains the interface pointer for the `Chat` object.

**HRESULT Remove ([in] long Index )**

This command removes an item from a `ChatCollection`.

**Parameters:**
- **[in] Index** contains the item to be removed.
Property Documentation

HRESULT Item [get]

This command queries a chat collection item with a specified index.

Parameters:

[in] Index contains the item to be returned.

The documentation for this interface was generated from the following file:

- Skype4COM.idl

Copyright © 2006 Skype Limited. All rights reserved.
Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td>Class List</td>
<td>Class Members</td>
</tr>
<tr>
<td>Properties</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
IChatMember Interface Reference

**ChatMember** class default interface. More...

import "Skype4COM.idl";

List of all members.
## Properties

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BSTR</td>
<td>Id [get]</td>
<td>Returns member id.</td>
</tr>
<tr>
<td>BSTR</td>
<td>Handle [get]</td>
<td>Returns member Skypename.</td>
</tr>
<tr>
<td>TChatMemberRole</td>
<td>Role [get, set]</td>
<td>Sets/returns member role.</td>
</tr>
<tr>
<td>VARIANT_BOOL</td>
<td>CanSetRoleTo ([in] TChatMemberRole Role) [get]</td>
<td>Checks if the new role can be applied to member.</td>
</tr>
<tr>
<td>VARIANT_BOOL</td>
<td>IsActive [get]</td>
<td>Returns member activity status.</td>
</tr>
<tr>
<td>IChat</td>
<td>Chat [get]</td>
<td>Returns related chat object.</td>
</tr>
</tbody>
</table>
Detailed Description

ChatMember class default interface.
Property Documentation

HRESULT CanSetRoleTo [get]

Checks if the new role can be applied to member.

Parameters:
[in] Role contains the new role.

The documentation for this interface was generated from the following file:

- Skype4COM.idl

Copyright � 2006 Skype Limited. All rights reserved.
ChatMemberCollection class default interface. More...

import "Skype4COM.idl";

List of all members.
### Properties

<table>
<thead>
<tr>
<th>Type</th>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LONG</td>
<td><strong>Count</strong> [get]</td>
<td>This command queries the number of chat members in a <strong>ChatMemberCollection</strong>.</td>
</tr>
<tr>
<td>IChatMember</td>
<td><strong>Item ([in] LONG Index) [get]</strong></td>
<td>This command queries an item in a <strong>ChatMemberCollection</strong> with a specified index.</td>
</tr>
<tr>
<td>IUnknown</td>
<td><strong>_NewEnum</strong> [get]**</td>
<td>This command queries an enumerator object for a <strong>ChatMemberCollection</strong>.</td>
</tr>
</tbody>
</table>
Detailed Description

ChatMemberCollection class default interface.
## Property Documentation

**HRESULT Item [get]**

This command queries an item in a `ChatMemberCollection` with a specified index.

**Parameters:**

- `[in] Index` contains the item to be returned.

The documentation for this interface was generated from the following file:

- Skype4COM.idl

---

Copyright © 2006 Skype Limited. All rights reserved.
<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Properties
ChatMessage class default interface. More...

import "Skype4COM.idl";

List of all members.
## Properties

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LONG</td>
<td><code>Id</code> [get]</td>
<td>This command queries a message ID.</td>
</tr>
<tr>
<td>DATE</td>
<td><code>Timestamp</code> [get]</td>
<td>This command queries a message timestamp.</td>
</tr>
<tr>
<td>BSTR</td>
<td><code>FromHandle</code> [get]</td>
<td>This command queries the Skypename of a message sender.</td>
</tr>
<tr>
<td>BSTR</td>
<td><code>FromDisplayName</code> [get]</td>
<td>This command queries the display name of a sender.</td>
</tr>
<tr>
<td>TChatMessageType</td>
<td><code>Type</code> [get]</td>
<td>This command queries a message type.</td>
</tr>
<tr>
<td>TChatMessageStatus</td>
<td><code>Status</code> [get]</td>
<td>This command queries a message status.</td>
</tr>
<tr>
<td>TChatLeaveReason</td>
<td><code>LeaveReason</code> [get]</td>
<td>This command queries the reason for leaving a chat.</td>
</tr>
<tr>
<td>BSTR</td>
<td><code>ChatName</code> [get]</td>
<td>This command queries the name of a chat that includes the message.</td>
</tr>
<tr>
<td>IUserCollection</td>
<td><code>Users</code> [get]</td>
<td>This command queries which users are added to chat.</td>
</tr>
<tr>
<td>VARIANT_BOOL</td>
<td><code>Seen</code> [set]</td>
<td>This command marks missed a chat message as seen.</td>
</tr>
<tr>
<td>IChat</td>
<td><code>Chat</code> [get]</td>
<td>This command queries a message chat.</td>
</tr>
<tr>
<td>IUser</td>
<td><code>Sender</code> [get]</td>
<td></td>
</tr>
</tbody>
</table>
This command queries the sender of a chat message.

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BSTR</td>
<td><strong>EditedBy</strong> [get]</td>
<td>Returns chat message editor Skypename.</td>
</tr>
<tr>
<td>DATE</td>
<td><strong>EditedTimestamp</strong> [get]</td>
<td>Returns chat message editing timestamp.</td>
</tr>
<tr>
<td>TChatMemberRole</td>
<td><strong>Role</strong> [get]</td>
<td>Returns chat member role changed by this message.</td>
</tr>
<tr>
<td>LONG</td>
<td><strong>Options</strong> [get]</td>
<td>Returns chat options changed by this message.</td>
</tr>
<tr>
<td>VARIANT_BOOL</td>
<td><strong>IsEditable</strong> [get]</td>
<td>Returns chat message editable status.</td>
</tr>
</tbody>
</table>
Detailed Description

ChatMessage class default interface.
Property Documentation

HRESULT LeaveReason [get]

This command queries the reason for leaving a chat.

**Note:**
Applies to messages with TChatMessageType::cmeLeft type.

HRESULT Options [get]

Returns chat options changed by this message.

**Note:**
Applies to messages with TChatMessageType::cmeSetOptions type.

HRESULT Role [get]

Returns chat member role changed by this message.

**Note:**
Applies to messages with TChatMessageType::cmeSetRole type.

The documentation for this interface was generated from the following file:

- Skype4COM.idl

Copyright ❤️ 2006 Skype Limited. All rights reserved.
## Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Public Member Functions
Properties
ChatMessageCollection class default interface. More...

import "Skype4COM.idl";

List of all members.
# Public Member Functions

<table>
<thead>
<tr>
<th>HRESULT</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add ([in] IChatMessage *pItem)</td>
<td>This command adds a new item to a ChatMessageCollection.</td>
<td></td>
</tr>
<tr>
<td>Remove ([in] long Index)</td>
<td>This command removes an item from a ChatMessageCollection.</td>
<td></td>
</tr>
<tr>
<td>RemoveAll (void)</td>
<td>This command removes all items from a ChatMessageCollection.</td>
<td></td>
</tr>
</tbody>
</table>
## Properties

<table>
<thead>
<tr>
<th>LONG</th>
<th>Count [get]</th>
</tr>
</thead>
<tbody>
<tr>
<td>This command queries the number of chat messages in a <strong>ChatMessageCollection</strong>.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IChatMessage</th>
<th>Item ([in] LONG Index) [get]</th>
</tr>
</thead>
<tbody>
<tr>
<td>This command queries an item in a <strong>ChatMessageCollection</strong> with a specified index.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IUnknown</th>
<th>_NewEnum [get]</th>
</tr>
</thead>
<tbody>
<tr>
<td>This command queries an enumerator object for a <strong>ChatMessageCollection</strong>.</td>
<td></td>
</tr>
</tbody>
</table>
Detailed Description

ChatMessageCollection class default interface.
Member Function Documentation

HRESULT Add ([in] IChatMessage * pItem )

This command adds a new item to a ChatMessageCollection.

Parameters:

HRESULT Remove ([in] long Index )

This command removes an item from a ChatMessageCollection.

Parameters:
  [in] Index contains the item to be removed.
Property Documentation

HRESULT Item [get]

This command queries an item in a `ChatMessageCollection` with a specified index.

Parameters:
  
  [in] **Index** contains the item to be returned.

The documentation for this interface was generated from the following file:

- **Skype4COM.idl**

Copyright 2006 Skype Limited. All rights reserved.
## Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Public Member Functions

### Properties
IClient Interface Reference

**Client** class default interface. [More...](#)

import "Skype4COM.idl";

List of all members.
## Public Member Functions

<table>
<thead>
<tr>
<th>HRESULT</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HRESULT</td>
<td><strong>Start</strong> ([in, defaultvalue(0)] VARIANT_BOOL Minimized,[in, defaultvalue(0)] VARIANT_BOOL Nosplash)</td>
<td>This command starts the Skype client.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>Minimize</strong> ()</td>
<td>This command minimizes (hides) the Skype client.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>Shutdown</strong> ()</td>
<td>This command closes the Skype client.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>OpenProfileDialog</strong> ()</td>
<td>This command opens the profile window for the current user.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>OpenUserInfoDialog</strong> ([in] BSTR Username)</td>
<td>This command opens the information window for the current user.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>OpenConferenceDialog</strong> ()</td>
<td>This command opens the create conference window.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>OpenSearchDialog</strong> ()</td>
<td>This command opens the search contacts window.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>OpenOptionsDialog</strong> ([in] BSTR Page)</td>
<td>This command opens the options window.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>OpenCallHistoryTab</strong> ()</td>
<td>This command brings the call history tab into focus.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>OpenContactsTab</strong> ()</td>
<td>This command brings the contacts tab into focus.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>OpenDialpadTab</strong> ()</td>
<td>This command brings the dialpad tab into focus.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>OpenSendContactsDialog</strong> ([in, defaultvalue(&quot;&quot;')] BSTR Username)</td>
<td>This command opens the send contacts window.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>OpenBlockedUsersDialog</strong> ()</td>
<td>This command opens the blocked users window.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>OpenImportContactsWizard</strong> ()</td>
<td>This command opens the import contacts window.</td>
</tr>
<tr>
<td>HRESULT</td>
<td>Function Call</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>---------------</td>
<td>-------------</td>
</tr>
<tr>
<td>HRESULT</td>
<td>OpenGettingStartedWizard ()</td>
<td>This command opens the getting started wizard.</td>
</tr>
<tr>
<td>HRESULT</td>
<td>OpenAuthorizationDialog ([in] BSTR Username)</td>
<td>This command opens the user authorization window.</td>
</tr>
<tr>
<td>HRESULT</td>
<td>OpenDialog ([in] BSTR Name,[in, defaultvalue('')] BSTR Param1,[in, defaultvalue('')] BSTR Param2)</td>
<td>This command opens the user interface window.</td>
</tr>
<tr>
<td>HRESULT</td>
<td>OpenVideoTestDialog ()</td>
<td>This command opens the video test window.</td>
</tr>
<tr>
<td>HRESULT</td>
<td>OpenAddContactDialog ([in, defaultvalue('')] BSTR Username)</td>
<td>This command opens the &quot;Add a Contact&quot; window.</td>
</tr>
<tr>
<td>HRESULT</td>
<td>OpenMessageDialog ([in] BSTR Username,[in, defaultvalue('')] BSTR Text)</td>
<td>This command opens the &quot;Send an IM Message&quot; window.</td>
</tr>
<tr>
<td>HRESULT</td>
<td>OpenFileTransferDialog ([in] BSTR User,[in] BSTR Folder)</td>
<td>This command opens the file transfer window.</td>
</tr>
<tr>
<td>HRESULT</td>
<td>Focus ()</td>
<td>This command brings the client window into focus.</td>
</tr>
<tr>
<td>HRESULT</td>
<td>ButtonPressed ([in] BSTR Key)</td>
<td>This command sends a button pressed notification event.</td>
</tr>
<tr>
<td>HRESULT</td>
<td>ButtonReleased ([in] BSTR Key)</td>
<td>This command sends a button released notification event.</td>
</tr>
<tr>
<td>HRESULT</td>
<td>OpenSmsDialog ([in] BSTR SmsId)</td>
<td>This command opens SMS window.</td>
</tr>
<tr>
<td>HRESULT</td>
<td>CreateEvent ([in] BSTR EventId,[in] BSTR Caption,[in] BSTR Hint,[out, retval] IPluginEvent **pEvent)</td>
<td>This command creates a plug-in event.</td>
</tr>
<tr>
<td>HRESULT</td>
<td>CreateMenuItem ([in] BSTR MenuItemId,[in] TPluginContext PluginContext,[in] BSTR CaptionText,[in, defaultvalue('')] BSTR HintText,[in, defaultvalue('')] BSTR IconPath,[in, defaultvalue(-1)] VARIANT_BOOL Enabled,[in, defaultvalue(pluginContactTypeAll)])</td>
<td></td>
</tr>
</tbody>
</table>
**TPluginContactType** ContactType,[in, defaultvalue(0)]
VARIANT_BOOL MultipleContacts,[out, retval]
IPluginMenuItem **pMenuItem)**
This command creates a menu item for a plug-in.

**HRESULT** **OpenLiveTab** ()
This command brings the Live tab into focus.
### Properties

<table>
<thead>
<tr>
<th>VARIANT_BOOL</th>
<th><strong>IsRunning</strong> [get]</th>
<th>This command queries the running status of the Skype client.</th>
</tr>
</thead>
<tbody>
<tr>
<td>BSTR</td>
<td><strong>Wallpaper</strong> [get, set]</td>
<td>This command queries/sets path to wallpaper bitmap.</td>
</tr>
</tbody>
</table>
Detailed Description

**Client** class default interface.
Member Function Documentation

HRESULT ButtonPressed ([in] BSTR Key)

This command sends a button pressed notification event.

Parameters:
[in] Key is a button key (0-9, A-Z, #, *, UP, DOWN, YES, NO, SKYPE, PAGEUP, PAGEDOWN).

HRESULT ButtonReleased ([in] BSTR Key)

This command sends a button released notification event.

Parameters:
[in] Key is button key (0-9, A-Z, #, *, UP, DOWN, YES, NO, SKYPE, PAGEUP, PAGEDOWN).


This command creates a plug-in event.

Parameters:
[in] EventId is a unique identifier for the plug-in event.
[in] Caption event contains caption text.
[in] Hint event contains hint text.
[out] pEvent receives the interface pointer for the PluginEvent object.
HRESULT CreateMenuItem (
	[in] BSTR MenuItemId,
	[in] TPluginContext PluginContext,
	[in] BSTR CaptionText,
	[in, defaultvalue(""')] BSTR HintText,
	[in, defaultvalue(""')] BSTR IconPath,
	[in, defaultvalue(-1)] VARIANT_BOOL Enabled,
	[in, defaultvalue(pluginContactTypeAll)] VARIANT_BOOL ContactType,
	[in, defaultvalue(0)] VARIANT_BOOL MultipleContacts,
	[out, retval] IPluginMenuItem ** pMenuItem)

This command creates a menu item for a plug-in.

Parameters:

- **[in] MenuItemId** is a unique identifier for the menu item.
- **[in] PluginContext** contains the plug-in context.
- **[in] CaptionText** contains the caption text for the menu item.
- **[in] HintText** contains the hint text for the menu item (optional).
- **[in] IconPath** is the path to an icon file (optional).
- **[in] Enabled** defines the initial state of the menu item (optional - default enabled).
- **[in] ContactType** is the contact type and is relevant only the TPluginContext::pluginContextContact context (optional - default all contacts).
- **[in] MultipleContacts** defines whether multiple contacts are allowed for this menu item (optional - default does not allow multiple contacts).
- **[out] pMenuItem** receives the interface pointer for the PluginMenuItem object.

See also:
- **TPluginContext**
HRESULT OpenAddContactDialog ([in, defaultvalue(""')] BSTR Username)

This command opens the "Add a Contact" window.

Parameters:
    [in] Username is an optional field for the Skypename of the contact.

HRESULT OpenAuthorizationDialog ([in] BSTR Username)

This command opens the user authorization window.

Parameters:
    [in] Username contains the Skypename of the user.

HRESULT OpenDialog ([in] BSTR Name,
                    [in, defaultvalue(""')] BSTR Param1, 
                    [in, defaultvalue(""')] BSTR Param2)

This command opens the user interface window.

Parameters:
    [in] Name The Name field contains the name of the window.
    [in] Param1 is an optional parameter.
    [in] Param2 is an optional parameter.

HRESULT OpenFileTransferDialog ([in] BSTR Username, 
                                [in] BSTR Folder)

This command opens the file transfer window.
Parameters:
- **Username** contains the Skypename of the user
- **Folder** contains the folder name to transfer.

HRESULT OpenMessageDialog ([in] BSTR Username
[in, defaultvalue(""')] BSTR Text)

This command opens the "Send an IM Message" window.

Parameters:
- **Username** contains the message target.
- **Text** is an optional field which can contain message text.

HRESULT OpenOptionsDialog ([in] BSTR Page)

This command opens the options window.

Parameters:
- **Page** contains the page to open.

HRESULT OpenSendContactsDialog ([in, defaultvalue(""')] BSTR Username)

This command opens the send contacts window.

Parameters:
- **Username** is an optional field for the Skypename of the user.

HRESULT OpenSmsDialog ([in] BSTR SmsId)
This command opens SMS window.

**Parameters:**

- **[in]** `SmsId` is SMS message id.

```
HRESULT OpenUserInfoDialog ([in] BSTR Username)
```

This command opens the information window for the current user.

**Parameters:**

- **[in]** `Username` contains the Skypename of the user.

```
HRESULT Start ([in, defaultvalue(0)] VARIANT_BOOL Minimized,
               [in, defaultvalue(0)] VARIANT_BOOL Nosplash)
```

This command starts the Skype client.

**Parameters:**

- **[in]** `Minimized` if True, Skype is minimized in the system tray.
- **[in]** `Nosplash` if True, Skype does not display a splash screen on start up.

The documentation for this interface was generated from the following file:

- Skype4COM.idl

Copyright © 2006 Skype Limited. All rights reserved.
### Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Properties
ICommand Interface Reference

**Command** class default interface. More...

import "Skype4COM.idl";

List of all members.
## Properties

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Get/Set</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LONG</td>
<td><strong>Id</strong></td>
<td>[get, set]</td>
<td>This command queries/sets a command ID.</td>
</tr>
<tr>
<td>LONG</td>
<td><strong>Timeout</strong></td>
<td>[get, set]</td>
<td>This command queries/sets the wait timeout in milliseconds.</td>
</tr>
<tr>
<td>VARIANT_BOOL</td>
<td><strong>Blocking</strong></td>
<td>[get, set]</td>
<td>This command queries/sets if blocking is set.</td>
</tr>
<tr>
<td>BSTR</td>
<td><strong>Command</strong></td>
<td>[get, set]</td>
<td>This command queries/sets the command text.</td>
</tr>
<tr>
<td>BSTR</td>
<td><strong>Reply</strong></td>
<td>[get, set]</td>
<td>This command queries/sets the reply text.</td>
</tr>
<tr>
<td>BSTR</td>
<td><strong>Expected</strong></td>
<td>[get, set]</td>
<td>This command queries/sets the expected reply text.</td>
</tr>
</tbody>
</table>
Detailed Description

**Command** class default interface.

The documentation for this interface was generated from the following file:

- **Skype4COM.idl**

Copyright © 2006 Skype Limited. All rights reserved.
Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Files</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class List</td>
<td>Class Members</td>
</tr>
</tbody>
</table>

Public Member Functions | Properties
Conference class default interface. More...

import "Skype4COM.idl";

List of all members.
## Public Member Functions

<table>
<thead>
<tr>
<th>HRESULT</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HRESULT</td>
<td><strong>Hold ()</strong></td>
<td>This command places all calls in a conference on hold and all active calls have the status <code>TCallStatus::clsLocalHold</code>.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>Resume ()</strong></td>
<td>This command resumes a conference that was placed on hold and all active calls have the status <code>TCallStatus::clsInProgress</code>.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>Finish ()</strong></td>
<td>This command finishes a conference and all active calls have the status <code>TCallStatus::clsFinished</code>.</td>
</tr>
</tbody>
</table>
## Properties

<table>
<thead>
<tr>
<th>Type</th>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LONG</td>
<td>Id [get]</td>
<td>This command queries the ID of a conference.</td>
</tr>
<tr>
<td>ICallCollection</td>
<td>Calls [get]</td>
<td>This command queries calls with the same conference ID.</td>
</tr>
<tr>
<td>ICallCollection</td>
<td>ActiveCalls [get]</td>
<td>This command queries active calls with the same conference ID.</td>
</tr>
</tbody>
</table>
Detailed Description

**Conference** class default interface.

The documentation for this interface was generated from the following file:

- Skype4COM.idl

Copyright © 2006 Skype Limited. All rights reserved.
# Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Files</strong></td>
<td><strong>Examples</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Public Member Functions | Properties
**ConferenceCollection** class default interface. More...

import "Skype4COM.idl";

List of all members.
## Public Member Functions

<table>
<thead>
<tr>
<th>HRESULT</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HRESULT</td>
<td><strong>Add</strong> ([in] IConference *pItem)</td>
<td>This command adds a new conference to a ConferenceCollection.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>Remove</strong> ([in] long Index)</td>
<td>This command removes an item from a ConferenceCollection.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>RemoveAll</strong> (void)</td>
<td>This command removes all items from a ConferenceCollection.</td>
</tr>
</tbody>
</table>
### Properties

<table>
<thead>
<tr>
<th>Type</th>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LONG</td>
<td><strong>Count</strong> [get]</td>
<td>This command queries the number of conferences in a ConferenceCollection.</td>
</tr>
<tr>
<td>IConference</td>
<td><strong>Item</strong> ([in] LONG Index) [get]</td>
<td>This command queries a ConferenceCollection item with a specified index.</td>
</tr>
<tr>
<td>IUnknown</td>
<td><strong>_NewEnum</strong> [get]</td>
<td>This command queries the enumerator object of a ConferenceCollection.</td>
</tr>
</tbody>
</table>
Detailed Description

ConferenceCollection class default interface.
Member Function Documentation

HRESULT Add ([in] IConference * pConference )

This command adds a new conference to a ConferenceCollection.

Parameters:
  [in] pConference receives the interface pointer for the Conference object.

HRESULT Remove ([in] long Index )

This command removes an item from a ConferenceCollection.

Parameters:
  [in] Index contains the item to be removed.
Property Documentation

HRESULT Item [get]

This command queries a ConferenceCollection item with a specified index.

Parameters:
[in] Index contains the item to be returned.

The documentation for this interface was generated from the following file:

- Skype4COM.idl

Copyright © 2006 Skype Limited. All rights reserved.
Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Public Member Functions | Properties
IConversion Interface Reference

**Conversion** class default interface. More...

import "Skype4COM.idl";

List of all members.
## Public Member Functions

<table>
<thead>
<tr>
<th>HRESULT</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HRESULT</td>
<td><strong>OnlineStatusToText</strong> ([in] TOnlineStatus status,[out, retval] BSTR *pVal)</td>
<td>This command converts a user's online status from status code to text.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>TextToOnlineStatus</strong> ([in] BSTR Text,[out, retval] TOnlineStatus *pVal)</td>
<td>This command converts text to online status code.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>BuddyStatusToText</strong> ([in] TBuddyStatus status,[out, retval] BSTR *pVal)</td>
<td>This command converts contact status to text.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>TextToBuddyStatus</strong> ([in] BSTR Text,[out, retval] TBuddyStatus *pVal)</td>
<td>This command converts contact status text to contact status code.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>CallStatusToText</strong> ([in] TCallStatus Status,[out, retval] BSTR *pVal)</td>
<td>This command converts call status to text.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>TextToCallStatus</strong> ([in] BSTR Text,[out, retval] TCallStatus *pVal)</td>
<td>This command converts call status text to call status code.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>CallTypeToText</strong> ([in] TCallType CallType,[out, retval] BSTR *pVal)</td>
<td>This command converts the call type to text.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>TextToCallType</strong> ([in] BSTR Text,[out, retval] TCallType *pVal)</td>
<td>This command converts call type text to call type code.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>UserSexToText</strong> ([in] TUserSex Sex,[out, retval] BSTR *pVal)</td>
<td>This command converts user sex value to text.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>TextToUserSex</strong> ([in] BSTR Text,[out, retval] TUserSex *pVal)</td>
<td></td>
</tr>
<tr>
<td>HRESULT</td>
<td>Function Name</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>---------------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>HRESULT</td>
<td>ConnectionStatusToText ([in] TConnectionStatus Status,[out, retval] BSTR *pVal)</td>
<td>This command converts the connection status to a text string.</td>
</tr>
<tr>
<td>HRESULT</td>
<td>TextToConnectionStatus ([in] BSTR Text,[out, retval] TConnectionStatus *pVal)</td>
<td>This command converts a text string of connection status to a code value.</td>
</tr>
<tr>
<td>HRESULT</td>
<td>UserStatusToText ([in] TUUserStatus Status,[out, retval] BSTR *pVal)</td>
<td>This command converts the user status value to a text string.</td>
</tr>
<tr>
<td>HRESULT</td>
<td>TextToUserStatus ([in] BSTR Text,[out, retval] TUUserStatus *pVal)</td>
<td>This command converts a string of user status to a code value.</td>
</tr>
<tr>
<td>HRESULT</td>
<td>CallFailureReasonToText ([in] TCallFailureReason reason,[out, retval] BSTR *pVal)</td>
<td>This command converts a failure reason to a text string.</td>
</tr>
<tr>
<td>HRESULT</td>
<td>AttachmentStatusToText ([in] TAttachmentStatus status,[out, retval] BSTR *pVal)</td>
<td>This command converts an attachment status code to a text string.</td>
</tr>
<tr>
<td>HRESULT</td>
<td>ChatLeaveReasonToText ([in] TChatLeaveReason reason,[out, retval] BSTR *pVal)</td>
<td>This command converts a leave reason to a text string.</td>
</tr>
<tr>
<td>HRESULT</td>
<td>ChatStatusToText ([in] TChatStatus status,[out, retval] BSTR *pVal)</td>
<td>This command converts a chat status code to a text string.</td>
</tr>
<tr>
<td>HRESULT</td>
<td>VoicemailTypeToText ([in] TVoicemailType type,[out, retval] BSTR *pVal)</td>
<td>This command converts a voicemail type to a text string.</td>
</tr>
<tr>
<td>HRESULT</td>
<td>VoicemailStatusToText ([in] TVoicemailStatus status, [out, retval] BSTR *pVal)</td>
<td>This command converts a voicemail status to a text string.</td>
</tr>
</tbody>
</table>
This command converts a voicemail status to a text string.

<table>
<thead>
<tr>
<th>HRESULT</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HRESULT</td>
<td><code>TextToVoicemailStatus</code> ([in] BSTR Text,[out, retval] TVoicemailStatus *pVal)</td>
<td>This command converts a voicemail status text string to code.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><code>VoicemailFailureReasonToText</code> ([in] TVoicemailFailureReason code,[out, retval] BSTR *pVal)</td>
<td>This command converts a voicemail failure reason to a text string.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><code>ChatMessageStatusToText</code> ([in] TChatMessageStatus Status,[out, retval] BSTR *pVal)</td>
<td>This command converts a message status code to text.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><code>TextToChatMessageStatus</code> ([in] BSTR Text,[out, retval] TChatMessageStatus *pVal)</td>
<td>This command converts a message status from text to code.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><code>ChatMessageTypeToText</code> ([in] TChatMessageType Type,[out, retval] BSTR *pVal)</td>
<td>This command converts the message type code to text.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><code>TextToChatMessageType</code> ([in] BSTR Text,[out, retval] TChatMessageType *pVal)</td>
<td>This command converts a message type from text string to code.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><code>TextToAttachmentStatus</code> ([in] BSTR Text,[out, retval] TAttachmentStatus *pVal)</td>
<td>This command converts an attachment status text string to code.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><code>GroupTypeToText</code> ([in] TGroupType Type,[out, retval] BSTR *pVal)</td>
<td>This command converts a group type to a text string.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><code>TextToGroupType</code> ([in] BSTR Text,[out, retval] TGroupType *pVal)</td>
<td>This command converts a text group type to a code value.</td>
</tr>
<tr>
<td>HRESULT</td>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>-----------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>HRESULT</td>
<td><code>CallVideoStatusToText</code> ([in] TCallVideoStatus Status, [out, retval] BSTR *pVal)</td>
<td>This command converts call video status to a text string.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><code>CallVideoSendStatusToText</code> ([in] TCallVideoSendStatus Status, [out, retval] BSTR *pVal)</td>
<td>This command converts call video send status to a text string.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><code>SmsMessageStatusToText</code> ([in] TSmsMessageStatus status, [out, retval] BSTR *pVal)</td>
<td>This command converts SMS message status to a text string.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><code>SmsMessageTypeToText</code> ([in] TSmsMessageType type, [out, retval] BSTR *pVal)</td>
<td>This command converts SMS message type to a text string.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><code>SmsTargetStatusToText</code> ([in] TSmsTargetStatus status, [out, retval] BSTR *pVal)</td>
<td>This command converts SMS target status to a text string.</td>
</tr>
</tbody>
</table>
### Properties

<table>
<thead>
<tr>
<th>BSTR Language [get, set]</th>
</tr>
</thead>
<tbody>
<tr>
<td>This command queries/sets the language to use for a text conversion.</td>
</tr>
</tbody>
</table>
Detailed Description

Conversion class default interface.

This interface converts Skype4COM constant values to text and converts text back to constant values.

See also:
Refer to the Skype4COM Dev Notes for more information about conversion.
Member Function Documentation

HRESULT AttachmentStatusToText ( [in] TAttachmentStatus status
 [out, retval] BSTR * pVal )

This command converts an attachment status code to a text string.

Parameters:
[ in] status contains the attachment status code.
[ out] pVal receives the converted attachment status as a text string.

HRESULT BuddyStatusToText ( [in] TBuddyStatus status,
 [out, retval] BSTR * pVal )

This command converts contact status to text.

Parameters:
[ in] status is the contact status value.
[ out] pVal receives the contact status value as a text string.

HRESULT CallFailureReasonToText ( [in] TCallFailureReason reason
 [out, retval] BSTR * pVal )

This command converts a failure reason to a text string.

Parameters:
[ in] reason contains a failure reason code.
[ out] pVal receives the failure reason as a text string.
HRESULT CallStatusToText ([in] TCallStatus status, [out, retval] BSTR * pVal)

This command converts call status to text.

Parameters:
[in] status contains the call status value.
[out] pVal receives the call status as a text string.

HRESULT CallTypeToText ([in] TCallType CallType, [out, retval] BSTR * pVal)

This command converts the call type to text.

Parameters:
[in] CallType contains the call type value.
[out] pVal receives the call type value as a text string.

HRESULT CallVideoSendStatusToText ([in] TCallVideoSendStatus Status, [out, retval] BSTR * pVal)

This command converts call video send status to a text string.

Parameters:
[in] Status contains the video send status code.
[out] pVal receives the video send status as a text string.

HRESULT CallVideoStatusToText ([in] TCallVideoStatus Status,
This command converts call video status to a text string.

**Parameters:**
- **Status**: contains the video status code.
- **pVal**: receives the converted code as a text string.

```c
HRESULT ChatMessageStatusToText ([in] TChatMessageStatus Status, [out, retval] BSTR * pVal)
```

This command converts a leave reason to a text string.

**Parameters:**
- **reason**: contains a leave reason code.
- **pVal**: receives a leave reason as a text string.

```c
HRESULT ChatLeaveReasonToText ([in] TChatLeaveReason reason, [out, retval] BSTR * pVal)
```

This command converts message status code to text.

**Parameters:**
- **Status**: contains the message status value.
- **pVal**: receives the message status as a text string.

```c
HRESULT ChatMessageTypeToText ([in] TChatMessageType Type, [out, retval] BSTR * pVal)
```
This command converts the message type code to text.

**Parameters:**
- `[in]` **Type** contains the message type code.
- `[out]` **pVal** receives the message type as a text string.

```c
HRESULT ChatStatusToText ([in] TChatStatus status, [out, retval] BSTR * pVal)
```

This command converts a chat status code to a text string.

**Parameters:**
- `[in]` **status** contains the chat status code.
- `[out]` **pVal** receives the converted chat status as a text string.

```c
HRESULT ConnectionStatusToText ([in] TConnectionStatus Status, [out, retval] BSTR * pVal)
```

This command converts the connection status to a text string.

**Parameters:**
- `[in]` **Status** contains the connection status value.
- `[out]` **pVal** receives the connection status value as a text string.

```c
HRESULT GroupTypeToText ([in] TGroupType Type, [out, retval] BSTR * pVal)
```

This command converts a group type to a text string.
Parameters:
   [in] Type contains the group type code.
   [out] pVal receives the group type as a text string.

HRESULT OnlineStatusToText ([in] TOnlineStatus status,
                           [out, retval] BSTR * pVal)

This command converts a user's online status from status code to text.

Parameters:
   [in] status is the online status value.
   [out] pVal receives a text string of the current online status.

HRESULT SmsMessageStatusToText ([in] TSmsMessageStatus status,
                                 [out, retval] BSTR * pVal)

This command converts SMS message status to a text string.

Parameters:
   [in] status contains SMS message status.
   [out] pVal receives the status as text string.

HRESULT SmsMessageTypeToText ([in] TSmsMessageType type,
                              [out, retval] BSTR * pVal)

This command converts SMS message type to a text string.

Parameters:
[in]  type contains SMS message type.
[out] pVal receives the type as text string.

HRESULT SmsTargetStatusToText ([in] TSmsTargetStatus status,
                                  [out, retval] BSTR * pVal)

This command converts SMS target status to a text string.

Parameters:
[in]  status contains SMS target status.
[out] pVal receives the status as text string.

HRESULT TextToAttachmentStatus ([in] BSTR Text,
                                  [out, retval] TAttachmentStatus pVal)

This command converts an attachment status text string to code.

Parameters:
[in]  Text contains a text string
       ("UNKNOWN","SUCCESS","PENDING","REFUSED",
       "NOTAVAILABLE","AVAILABLE")
[out] pVal receives the converted text attachment status as a status code.

HRESULT TextToBuddyStatus ([in] BSTR Text,
                            [out, retval] TBuddyStatus * pVal)

This command converts contact status text to contact status code.

Parameters:
[in]  Text contains a text string of the contact status.
[out] pVal receives the contact status code.
HRESULT TextToCallStatus ([in] BSTR Text, [out, retval] TCallStatus * pVal)

This command converts call status text to call status code.

Parameters:
- [out] **pVal** receives the call status text as a call status code.

HRESULT TextToCallType ([in] BSTR Text, [out, retval] TCallType * pVal)

This command converts call type text to call type code.

Parameters:
- [in] **Text** contains a text string ("INCOMING_PSTN", "OUTGOING_PSTN", "INCOMING_P2P", "OUTGOING_P2P").
- [out] **pVal** receives the call type as call type code.

HRESULT TextToChatMessageStatus ([in] BSTR Text, [out, retval] TChatMessageType )

This command converts a message status from text to code.

Parameters:
[in]  **Text** contains a text string of the message status.
[out]  **pVal** receives the text string as message status code.

**HRESULT TextToChatMessageType ([in] BSTR Text, [out, retval] TChatMessageType * pVal)***

This command converts a message type from text string to code.

**Parameters:**

- **[in] Text** contains a text string for the message type.
- **[out] pVal** receives the message type in code.

**HRESULT TextToConnectionStatus ([in] BSTR Text, [out, retval] TConnectionStatus * pVal)***

This command converts a text string of connection status to a code value.

**Parameters:**

- **[in] Text** contains a text string ("OFFLINE", "CONNECTING", "PAUSING", "ONLINE", "LOGGEDOUT").
- **[out] pVal** receives the converted text string as connection status code.

**HRESULT TextToGroupType ([in] BSTR Text, [out, retval] TGroupType * pVal)***

This command converts a text group type to a code value.
### Parameters:


**[out] pVal** receives the converted text group type as code.

### HRESULT TextToOnlineStatus ([in] BSTR Text, [out, retval] TOnlineStatus * pVal)

This command converts text to online status code.

**Parameters:**

**[in] Text** contains a text string ("OFFLINE", "ONLINE", "AWAY", "NA", "DND", "SKYPEME", "SKYPEOUT").

**[out] pVal** receives the online status code.

### HRESULT TextToUserSex ([in] BSTR Text, [out, retval] TUserSex * pVal)

This command converts the user sex text string to code.

**Parameters:**

**[in] Text** contains a text string ("UNKNOWN", "MALE", "FEMALE")
"FEMALE"). [out]  

pVal receives the converted text string as user sex code.

HRESULT TextToUserStatus ([in] BSTR Text,  
[out, retval] TUserStatus * pVal
)

This command converts a string of user status to a code value.

**Parameters:**

[in]  

Text contains a text string ("OFFLINE", "ONLINE", "AWAY", "NA", "DND", "INVISIBLE", "LOGGEDOUT", "SKYPEME").

[out]  

pVal receives the converted text string as a user status code.

HRESULT TextToVoicemailStatus ([in] BSTR Text,  
[out, retval] TVoicemailStatus * pVal
)

This command converts a voicemail status text string to code.

**Parameters:**

[in]  


[out]  

pVal receives the converted voicemail status text as code.

HRESULT UserSexToText ([in] TUserSex Sex,
This command converts user sex value to text.

**Parameters:**

- **[in]** `Sex` contains the user sex value.
- **[out]** `pVal` receives the user sex value as a text string.

```c
HRESULT UserStatusToText ([in] TUserStatus Status,
                           [out, retval] BSTR * pVal
)
```

This command converts the user status value to a text string.

**Parameters:**

- **[in]** `Status` contains the user status value.
- **[out]** `pVal` receives the user status value as a text string.

```c
HRESULT VoicemailFailureReasonToText ([in] TVoicemailFailureReason Reason,
                                         [out, retval] BSTR * pVal
)
```

This command converts a voicemail failure reason to a text string.

**Parameters:**

- **[in]** `code` contains the voicemail failure reason code.
- **[out]** `pVal` receives the converted voicemail failure reason as a text string.

```c
HRESULT VoicemailStatusToText ([in] TVoicemailStatus status,
                                [out, retval] BSTR * pVal
)
```
This command converts a voicemail status to a text string.

**Parameters:**
- **[in]** `status` contains the voicemail status code.
- **[out]** `pVal` receives the converted voicemail status as a text string.

```
HRESULT VoicemailTypeToText ( [in] TVoicemailType type,
                             [out, retval] BSTR * pVal
)
```

This command converts a voicemail type to a text string.

**Parameters:**
- **[in]** `type` contains the voicemail type code.
- **[out]** `pVal` receives the converted voicemail type as a text string.

The documentation for this interface was generated from the following file:

- **Skype4COM.idl**

Copyright © 2006 Skype Limited. All rights reserved.
Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Properties</td>
</tr>
</tbody>
</table>
IFileTransfer Interface Reference

File transfer object default interface. More...

import "Skype4COM.idl";

List of all members.
# Properties

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BSTR</td>
<td><strong>Id</strong> [get]</td>
<td>This property returns the file transfer unique identifier.</td>
</tr>
<tr>
<td><strong>TFileTransferType</strong></td>
<td><strong>Type</strong> [get]</td>
<td>This property returns the file transfer type.</td>
</tr>
<tr>
<td><strong>TFileTransferStatus</strong></td>
<td><strong>Status</strong> [get]</td>
<td>This property returns the file transfer status.</td>
</tr>
<tr>
<td><strong>TFileTransferFailureReason</strong></td>
<td><strong>FailureReason</strong> [get]</td>
<td>This property returns the file transfer failure reason code.</td>
</tr>
<tr>
<td>BSTR</td>
<td><strong>PartnerHandle</strong> [get]</td>
<td>This property returns the file transfer partner skypename.</td>
</tr>
<tr>
<td>BSTR</td>
<td><strong>PartnerDisplayName</strong> [get]</td>
<td>This property returns the file transfer partner display name.</td>
</tr>
<tr>
<td>DATE</td>
<td><strong>StartTime</strong> [get]</td>
<td>This property returns the file transfer start timestamp.</td>
</tr>
<tr>
<td>DATE</td>
<td><strong>FinishTime</strong> [get]</td>
<td>This property returns the file transfer end timestamp.</td>
</tr>
<tr>
<td>BSTR</td>
<td><strong>FilePath</strong> [get]</td>
<td>This property returns the full path of the file being transferred.</td>
</tr>
<tr>
<td>BSTR</td>
<td><strong>FileName</strong> [get]</td>
<td>This property returns the file name of transferred file.</td>
</tr>
<tr>
<td>BSTR</td>
<td><strong>BytesPerSecond</strong> [get]</td>
<td>This property returns the transfer speed in bytes per second.</td>
</tr>
</tbody>
</table>
BSTR  **BytesTransferred**  [get]
This property returns the number of bytes transferred.
Detailed Description

File transfer object default interface.

The documentation for this interface was generated from the following file:

- Skype4COM.idl

Copyright © 2006 Skype Limited. All rights reserved.
## Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td>Public Member Functions</td>
<td>Properties</td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
IFileTransferCollection Interface Reference

File transfer collection class default interface. More...

import "Skype4COM.idl";

List of all members.
### Public Member Functions

<table>
<thead>
<tr>
<th>HRESULT</th>
<th>Member Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HRESULT</td>
<td><strong>Add</strong> ([in] IFileTransfer *plItem)</td>
<td>Adds a new item to a collection.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>Remove</strong> ([in] long Index)</td>
<td>This command removes an item from a collection.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>RemoveAll</strong> (void)</td>
<td>This command removes all items from a collection.</td>
</tr>
</tbody>
</table>
### Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>LONG</strong></td>
<td><strong>Count</strong> [get]</td>
</tr>
<tr>
<td><strong>IFileTransfer</strong></td>
<td><strong>Item ([in] LONG Index)</strong> [get]</td>
</tr>
<tr>
<td><strong>IUnknown</strong></td>
<td><strong>_NewEnum</strong> [get]</td>
</tr>
</tbody>
</table>
Detailed Description

File transfer collection class default interface.
Member Function Documentation

HRESULT Add ([in] IFileTransfer * pItem )

Adds a new item to a collection.

Parameters:
   [in] pItem contains the interface pointer for the File Transfer object.

HRESULT Remove ([in] long Index )

This command removes an item from a collection.

Parameters:
   [in] Index contains the item to be removed.
Property Documentation

HRESULT Item [get]

This command queries an item in a collection with a specified index.

Parameters:

[in] Index contains the item to be returned.

The documentation for this interface was generated from the following file:

- Skype4COM.idl

Copyright © 2006 Skype Limited. All rights reserved.
## Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Public Member Functions | Properties
IGroup Interface Reference

**Group** class default interface. More...

import "Skype4COM.idl";

List of all members.
# Public Member Functions

<table>
<thead>
<tr>
<th>HRESULT</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HRESULT</td>
<td><strong>AddUser</strong> ([in] BSTR Username)</td>
<td>This command adds a user to a group.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>RemoveUser</strong> ([in] BSTR Username)</td>
<td>This command removes a user from a group.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>Share</strong> ([in, defaultvalue(&quot;&quot;)] BSTR MessageText)</td>
<td>This command shares a contact group.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>Accept</strong> ()</td>
<td>This command accepts an invitation to join a shared contact group.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>Decline</strong> ()</td>
<td>This command declines an invitation to join a shared contact group.</td>
</tr>
</tbody>
</table>
## Properties

<table>
<thead>
<tr>
<th>Type</th>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LONG</td>
<td><strong>Id</strong> [get]</td>
<td>This command queries the group ID.</td>
</tr>
<tr>
<td>TGroupType</td>
<td><strong>Type</strong> [get]</td>
<td>This command queries a group type.</td>
</tr>
<tr>
<td>BSTR</td>
<td><strong>CustomGroupId</strong> [get]</td>
<td>This command queries a group persistent ID.</td>
</tr>
<tr>
<td>BSTR</td>
<td><strong>DisplayName</strong> [get, set]</td>
<td>This command queries/sets the display name of a group.</td>
</tr>
<tr>
<td>IUserCollection</td>
<td><strong>Users</strong> [get]</td>
<td>This command queries the users in a group.</td>
</tr>
<tr>
<td>IUserCollection</td>
<td><strong>OnlineUsers</strong> [get]</td>
<td>This command queries the online users in a group.</td>
</tr>
<tr>
<td>VARIANT_BOOL</td>
<td><strong>IsVisible</strong> [get]</td>
<td>This command queries the visibility status of a group.</td>
</tr>
<tr>
<td>VARIANT_BOOL</td>
<td><strong>IsExpanded</strong> [get]</td>
<td>This command queries the expanded status of a group.</td>
</tr>
</tbody>
</table>
Detailed Description

**Group** class default interface.
Member Function Documentation

HRESULT AddUser ([in] BSTR Username )

This command adds a user to a group.

Parameters:
  [in] Username contains the Skypename of the contact to add.

HRESULT RemoveUser ([in] BSTR Username )

This command removes a user from a group.

Parameters:
  [in] Username contains the Skypename of the contact to remove.

HRESULT Share ([in, defaultvalue(""')] BSTR MessageText )

This command shares a contact group.

Parameters:
Property Documentation

HRESULT CustomGroupId [get]

This command queries a group persistent ID.
The custom group ID is a persistent value that does not change.

The documentation for this interface was generated from the following file:

- Skype4COM.idl

Copyright � 2006 Skype Limited. All rights reserved.
# Skype4COM 1.0.36.0

## Classes

<table>
<thead>
<tr>
<th>Files</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class List</td>
<td>Class Members</td>
</tr>
</tbody>
</table>

## Public Member Functions

## Properties
**IGroupCollection Interface Reference**

*GroupCollection* class default interface. More...

import "Skype4COM.idl";

List of all members.
## Public Member Functions

<table>
<thead>
<tr>
<th>HRESULT</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add (in IGroup *pGroup)</td>
<td>This command adds a new item to a <strong>GroupCollection</strong>.</td>
<td></td>
</tr>
<tr>
<td>Remove (in long Index)</td>
<td>This command removes an item from a <strong>GroupCollection</strong>.</td>
<td></td>
</tr>
<tr>
<td>RemoveAll (void)</td>
<td>This command removes all items from a <strong>GroupCollection</strong>.</td>
<td></td>
</tr>
</tbody>
</table>
## Properties

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LONG</td>
<td>Count</td>
<td>[get] This command queries the number of groups in a GroupCollection.</td>
</tr>
<tr>
<td>IGroup</td>
<td>Item ([in] LONG Index)</td>
<td>[get] This command queries a GroupCollection item with a specified index.</td>
</tr>
<tr>
<td>IUnknown</td>
<td>_NewEnum</td>
<td>[get] This command queries a GroupCollection enumerator object.</td>
</tr>
</tbody>
</table>
Detailed Description

*GroupCollection* class default interface.
Member Function Documentation

**HRESULT Add ([in] IGroup * pGroup )**

This command adds a new item to a `GroupCollection`.

**Parameters:**

- **[in] pGroup** contains the interface pointer for the `Group` object.

**HRESULT Remove ([in] long Index )**

This command removes an item from a `GroupCollection`.

**Parameters:**

- **[in] Index** contains the item to be removed.
Property Documentation

HRESULT Item [get]

This command queries a GroupCollection item with a specified index.

Parameters:
[in] Index contains the item to be returned.

The documentation for this interface was generated from the following file:

- Skype4COM.idl

Copyright © 2006 Skype Limited. All rights reserved.
Skype4COM 1.0.36.0

| Main Page | Related Pages | Modules | **Classes** | Files | Examples | Class List | Class Members | Properties |
|-----------|---------------|---------|-------------|-------|----------|------------|---------------|------------|------------|
Participant class default interface. More...

import "Skype4COM.idl";

List of all members.
Properties

<table>
<thead>
<tr>
<th>Type</th>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BSTR</td>
<td>Handle [get]</td>
<td>This command queries the Skypename of a participant in a conference call.</td>
</tr>
<tr>
<td>BSTR</td>
<td>DisplayName [get]</td>
<td>This command queries the display name of a participant in a conference call.</td>
</tr>
<tr>
<td>TCallType</td>
<td>CallType [get]</td>
<td>This command queries a call type in a conference call.</td>
</tr>
<tr>
<td>TCallStatus</td>
<td>CallStatus [get]</td>
<td>This command queries the call status of a participant in a conference call.</td>
</tr>
</tbody>
</table>
Detailed Description

**Participant** class default interface.

The documentation for this interface was generated from the following file:

- **Skype4COM.idl**

Copyright © 2006 Skype Limited. All rights reserved.
## Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Files</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class List</td>
<td>Class Members</td>
</tr>
</tbody>
</table>

**Public Member Functions**
- Properties
**ParticipantCollection** class default interface. More...

```java
import "Skype4COM.idl";

List of all members.
### Public Member Functions

<table>
<thead>
<tr>
<th>HRESULT</th>
<th><strong>Add</strong> ([in] IParticipant *pParticipant)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>This command adds a new participant to a ParticipantCollection.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>HRESULT</th>
<th><strong>Remove</strong> ([in] long Index)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>This command removes an item from a ParticipantCollection.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>HRESULT</th>
<th><strong>RemoveAll</strong> (void)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>This command removes all items from a ParticipantCollection.</td>
</tr>
</tbody>
</table>
## Properties

<table>
<thead>
<tr>
<th>Type</th>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LONG</td>
<td><strong>Count</strong> [get]</td>
<td>This command queries the number of participants in a ParticipantCollection.</td>
</tr>
<tr>
<td>IParticipant</td>
<td><strong>Item</strong> ([in] LONG Index) [get]</td>
<td>This command queries an item with a specified index in a ParticipantCollection.</td>
</tr>
<tr>
<td>IUnknown</td>
<td><strong>_NewEnum</strong> [get]</td>
<td>This command queries the enumerator object of a ParticipantCollection.</td>
</tr>
</tbody>
</table>
Detailed Description

*ParticipantCollection* class default interface.
Member Function Documentation

HRESULT Add ([in] IParticipant * pParticipant )

This command adds a new participant to a ParticipantCollection.

Parameters:
    [in] pParticipant contains the interface pointer for the Participant object.

HRESULT Remove ([in] long Index )

This command removes an item from a ParticipantCollection.

Parameters:
    [in] Index contains the item to be removed.
Property Documentation

HRESULT Item [get]

This command queries an item with a specified index in a ParticipantCollection.

Parameters:

[in] Index contains the item to be returned.

The documentation for this interface was generated from the following file:

- Skype4COM.idl

Copyright � 2006 Skype Limited. All rights reserved.
<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Public Member Functions
Properties
IPluginEvent Interface Reference

This is the default interface for the PluginEvent class. More...

```java
import "Skype4COM.idl";

List of all members.
<table>
<thead>
<tr>
<th>HRESULT</th>
<th>Delete (void)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>This command deletes a plug-in event from the events pane in the Skype client.</td>
</tr>
</tbody>
</table>
# Properties

<table>
<thead>
<tr>
<th>BSTR</th>
<th>Id [get]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>This property is a unique identifier for a plug-in event.</td>
</tr>
</tbody>
</table>
Detailed Description

This is the default interface for the **PluginEvent** class.

The documentation for this interface was generated from the following file:

- **Skype4COM.idl**

Copyright © 2006 Skype Limited. All rights reserved.
Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Public Member Functions | Properties
Class **PluginMenuItem** default interface. More...

```
import "Skype4COM.idl";
```

List of all members.
**Public Member Functions**

<table>
<thead>
<tr>
<th>HRESULT</th>
<th>Delete (void)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>This command removes a menu item from &quot;Do More&quot; menus.</td>
</tr>
</tbody>
</table>
## Properties

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BSTR</td>
<td><strong>Id</strong> [get]</td>
<td>This property is a unique identifier for a plug-in menu item.</td>
</tr>
<tr>
<td>BSTR</td>
<td><strong>Caption</strong> [set]</td>
<td>This property contains the name of a plug-in menu item and must be a maximum of 32 characters.</td>
</tr>
<tr>
<td>BSTR</td>
<td><strong>Hint</strong> [set]</td>
<td>This property contains a second line of text for a plug-in menu item if required and must be a maximum of 32 characters.</td>
</tr>
<tr>
<td>VARIANT_BOOL</td>
<td><strong>Enabled</strong> [set]</td>
<td>This property is a Boolean indicator that defines whether a menu item is enabled when a user launches the application. If no value is defined, the menu item will be enabled.</td>
</tr>
</tbody>
</table>
Detailed Description

Class **PluginMenuItem** default interface.

The documentation for this interface was generated from the following file:

- **Skype4COM.idl**

Copyright © 2006 Skype Limited. All rights reserved.
## Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Properties
IPROfile Interface Reference

Profile class default interface. More...

import "Skype4COM.idl";

List of all members.
### Properties

<table>
<thead>
<tr>
<th>Type</th>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BSTR</td>
<td><strong>FullName</strong></td>
<td>[get, set] This command queries/sets the full name in the current user profile.</td>
</tr>
<tr>
<td>BSTR</td>
<td><strong>Birthday</strong></td>
<td>[get, set] This command queries/sets the birthday in the current user profile.</td>
</tr>
<tr>
<td>TUserSex</td>
<td><strong>Sex</strong></td>
<td>[get, set] This command queries/sets the sex in the current user profile.</td>
</tr>
<tr>
<td>BSTR</td>
<td><strong>Languages</strong></td>
<td>[get, set] This command queries/sets the language code in the current user profile.</td>
</tr>
<tr>
<td>BSTR</td>
<td><strong>Country</strong></td>
<td>[get, set] This command queries/sets the ISO country code in the current user profile.</td>
</tr>
<tr>
<td>BSTR</td>
<td><strong>Province</strong></td>
<td>[get, set] This command queries/sets the province in the current user profile.</td>
</tr>
<tr>
<td>BSTR</td>
<td><strong>City</strong></td>
<td>[get, set] This command queries/sets the city in the current user profile.</td>
</tr>
<tr>
<td>BSTR</td>
<td><strong>PhoneHome</strong></td>
<td>[get, set] This command queries/sets the home phone number in the current user profile.</td>
</tr>
<tr>
<td>BSTR</td>
<td><strong>PhoneOffice</strong></td>
<td>[get, set] This command queries/sets the business phone number in the current user profile.</td>
</tr>
<tr>
<td>BSTR</td>
<td><strong>PhoneMobile</strong></td>
<td>[get, set] This command queries/sets the mobile phone number in the current user profile.</td>
</tr>
<tr>
<td>BSTR</td>
<td><strong>Homepage</strong></td>
<td>[get, set] This command queries/sets the homepage url in the current user profile.</td>
</tr>
<tr>
<td>Type</td>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>---------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>BSTR</td>
<td><strong>About</strong> [get, set]</td>
<td>This command queries/sets the &quot;About&quot; text in the current user profile.</td>
</tr>
<tr>
<td>BSTR</td>
<td><strong>MoodText</strong> [get, set]</td>
<td>This command queries/sets the mood text in the current user profile.</td>
</tr>
<tr>
<td>LONG</td>
<td><strong>Timezone</strong> [get, set]</td>
<td>This command queries/sets the timezone in the current user profile.</td>
</tr>
<tr>
<td>short</td>
<td><strong>CallNoAnswerTimeout</strong> [get, set]</td>
<td>This command queries/sets the &quot;call no answer timeout&quot; in the current user profile.</td>
</tr>
<tr>
<td>VARIANT_BOOL</td>
<td><strong>CallApplyCF</strong> [get, set]</td>
<td>This command queries/sets if call forwarding is set in the current user profile.</td>
</tr>
<tr>
<td>VARIANT_BOOL</td>
<td><strong>CallSendToVM</strong> [get, set]</td>
<td>This command queries/sets whether calls will be sent to voicemail in the current user profile.</td>
</tr>
<tr>
<td>BSTR</td>
<td><strong>CallForwardRules</strong> [get, set]</td>
<td>This command queries/sets the call forwarding rules in the current user profile.</td>
</tr>
<tr>
<td>LONG</td>
<td><strong>Balance</strong> [get]</td>
<td>This command queries the balance in currency cents in the current user profile.</td>
</tr>
<tr>
<td>BSTR</td>
<td><strong>BalanceCurrency</strong> [get]</td>
<td>This command queries the currency code in the current user profile.</td>
</tr>
<tr>
<td>BSTR</td>
<td><strong>BalanceToText</strong> [get]</td>
<td>This command queries the balance amount with currency symbol in the current user profile.</td>
</tr>
<tr>
<td>BSTR</td>
<td><strong>IPCountry</strong> [get]</td>
<td>This command queries the ISO country code by IP address.</td>
</tr>
<tr>
<td>BSTR</td>
<td><strong>ValidatedSmsNumbers</strong> [get]</td>
<td>This command requests a list of validated SMS numbers.</td>
</tr>
<tr>
<td>BSTR</td>
<td><strong>RichMoodText</strong> [get, set]</td>
<td></td>
</tr>
<tr>
<td>-------</td>
<td>-----------------------------</td>
<td></td>
</tr>
<tr>
<td></td>
<td>This property returns/sets rich mood text in logged in user profile.</td>
<td></td>
</tr>
</tbody>
</table>
Detailed Description

Profile class default interface.
Property Documentation

HRESULT CallNoAnswerTimeout [get, set]

This command queries/sets the "call no answer timeout" in the current user profile.

The call no answer timeout is the period of time a call will ring without being answered before it stops ringing.

The documentation for this interface was generated from the following file:

- Skype4COM.idl

Copyright © 2006 Skype Limited. All rights reserved.
Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Public Member Functions | Properties
**ISettings Interface Reference**

*Settings* class default interface. More...

```plaintext
import "Skype4COM.idl";
```

List of all members.
Public Member Functions

<table>
<thead>
<tr>
<th>HRESULT</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HRESULT</td>
<td><strong>SaveAvatarToFile</strong> ([in] BSTR Filename,[in, defaultvalue(&quot;1&quot;)]) BSTR AvatarId</td>
<td>Saves current user avatar to file.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>LoadAvatarFromFile</strong> ([in] BSTR Filename,[in, defaultvalue(&quot;1&quot;)]) BSTR AvatarId</td>
<td>Replaces user current avatar with new avatar from file.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>ResetIdleTimer</strong> (void)</td>
<td>Reset Skype idle timer.</td>
</tr>
</tbody>
</table>
## Properties

<table>
<thead>
<tr>
<th>Type</th>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BSTR</td>
<td><strong>AudioIn</strong> [get, set]</td>
<td>This command queries/sets the name of an audio input device.</td>
</tr>
<tr>
<td>BSTR</td>
<td><strong>AudioOut</strong> [get, set]</td>
<td>This command queries/sets the name of an audio output device.</td>
</tr>
<tr>
<td>VARIANT_BOOL</td>
<td><strong>AutoAway</strong> [get, set]</td>
<td>This command queries/sets the &quot;Auto away&quot; status.</td>
</tr>
<tr>
<td>BSTR</td>
<td><strong>Ringer</strong> [get, set]</td>
<td>This command queries/sets the name of a ringer device.</td>
</tr>
<tr>
<td>BSTR</td>
<td><strong>Avatar</strong> ([in, defaultvalue(&quot;1&quot;)]) BSTR Id) [set]</td>
<td>This command sets the avatar picture in a user profile.</td>
</tr>
<tr>
<td>VARIANT_BOOL</td>
<td><strong>RingToneStatus</strong> ([in, defaultvalue(&quot;1&quot;)]) BSTR Id) [get, set]</td>
<td>This command queries or sets the ringtone status.</td>
</tr>
<tr>
<td>BSTR</td>
<td><strong>RingTone</strong> ([in, defaultvalue(&quot;1&quot;)]) BSTR Id) [set]</td>
<td>This command sets a new ringtone.</td>
</tr>
<tr>
<td>BSTR</td>
<td><strong>VideoIn</strong> [get, set]</td>
<td>This command queries/sets the name of a video input device.</td>
</tr>
<tr>
<td>VARIANT_BOOL</td>
<td><strong>PCSpeaker</strong> [get, set]</td>
<td>This command queries/sets the PC speaker status.</td>
</tr>
<tr>
<td>VARIANT_BOOL</td>
<td><strong>AGC</strong> [get, set]</td>
<td>This command queries/sets the status of automatic gain control.</td>
</tr>
<tr>
<td>VARIANT_BOOL</td>
<td><strong>AEC</strong> [get, set]</td>
<td>This command queries/sets the status of automatic echo cancellation.</td>
</tr>
<tr>
<td>BSTR</td>
<td><strong>Language</strong> [get, set]</td>
<td></td>
</tr>
</tbody>
</table>
This command queries/sets the language of the Skype client.
Detailed Description

Settings class default interface.
Member Function Documentation

HRESULT LoadAvatarFromFile ([in] BSTR Filename [in, defaultvalue("1")] BSTR AvatarId )

Replaces user current avatar with new avatar from file.

Parameters:
[in] Filename contains file name.
[in] AvatarId avatar id.

Example:
Set oSkype = CreateObject("Skype4COM.Skype")
oSkype.Settings.LoadAvatarFromFile("c:\temp\new_avatar.jpg")

HRESULT SaveAvatarToFile ([in] BSTR Filename [in, defaultvalue("1")] BSTR AvatarId )

Saves current user avatar to file.

Parameters:
[in] Filename contains file name.
[in] AvatarId avatar id.

Example:
Set oSkype = CreateObject("Skype4COM.Skype")
oSkype.Settings.SaveAvatarToFile("c:\temp\old_avatar.jpg")
Property Documentation

HRESULT Avatar [set]

This command sets the avatar picture in a user profile.

Parameters:
[in] Id is an optional avatar ID.
[in] newVal contains the new file name.

Deprecated:
Use the ISettings::LoadAvatarFromFile instead.

HRESULT RingTone [set]

This command sets a new ringtone.

Parameters:
[in] Id contains the ringtone ID.

HRESULT RingToneStatus [get, set]

This command queries or sets the ringtone status.

Parameters:
[in] Id contains the ringtone ID.

The documentation for this interface was generated from the following file:

- Skype4COM.idl
# Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Public Member Functions | Properties
**ISkype Interface Reference**

**Skype** is the default interface for the class. More...

```plaintext
import "Skype4COM.idl";

List of all members.
```
## Public Member Functions

<table>
<thead>
<tr>
<th>HRESULT</th>
<th>Function</th>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HRESULT</td>
<td><code>SearchForUsers</code></td>
<td>([in] BSTR Target,[out, retval] IUserCollection **pVal)</td>
<td>This command searches for users.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><code>Attach</code></td>
<td>([in, defaultvalue(5)] LONG Protocol,[in, defaultvalue(-1)] VARIANT_BOOL Wait)</td>
<td>This command establishes a connection to Skype.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><code>PlaceCall</code></td>
<td>([in] BSTR Target,[in, defaultvalue(&quot;&quot; )] BSTR Target2,[in, defaultvalue(&quot;&quot; )] BSTR Target3,[in, defaultvalue(&quot;&quot; )] BSTR Target4,[out, retval] ICall **pCall)</td>
<td>This command places a call to a single user or creates a conference call.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><code>SendMessage</code></td>
<td>([in] BSTR Username,[in] BSTR Text,[out, retval] IChatMessage **pMessage)</td>
<td>This command sends a chat message.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><code>SendCommand</code></td>
<td>([in] ICommand *pCommand)</td>
<td>This command sends an API command.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><code>ChangeUserStatus</code></td>
<td>([in] TUserStatus newVal)</td>
<td>This command changes the online status for the current user.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><code>CreateChatWith</code></td>
<td>([in] BSTR Username,[out, retval] IChat **pVal)</td>
<td>This command creates a chat with a single user.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><code>CreateChatMultiple</code></td>
<td>([in] IUserCollection *pMembers,[out, retval] IChat **pChat)</td>
<td>This command creates a chat with multiple users.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><code>SendVoicemail</code></td>
<td>([in] BSTR Username,[out, retval] IVoicemail **pVoicemail)</td>
<td>This command sends a voicemail to a specified user.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><code>ClearChatHistory</code></td>
<td>()</td>
<td>This command clears chat history.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><code>ClearVoicemailHistory</code></td>
<td>()</td>
<td>This command clears voicemail history.</td>
</tr>
<tr>
<td>HRESULT</td>
<td>Function</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>----------</td>
<td>-----------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>HRESULT</td>
<td>ClearCallHistory ([in, defaultvalue(&quot;ALL&quot;)]) BSTR Username,[in, defaultvalue(chsAllCalls)] TCallHistory type)</td>
<td>This command clears call history.</td>
<td></td>
</tr>
<tr>
<td>HRESULT</td>
<td>ResetCache ()</td>
<td>This command deletes all cache entries.</td>
<td></td>
</tr>
<tr>
<td>HRESULT</td>
<td>CreateGroup ([in] BSTR GroupName,[out, retval] IGroup **pGroup)</td>
<td>This command creates a contact group.</td>
<td></td>
</tr>
<tr>
<td>HRESULT</td>
<td>DeleteGroup ([in] LONG GroupId)</td>
<td>This command deletes a contact group. Users in the contact group are moved to the All Contacts hardwired contact group.</td>
<td></td>
</tr>
<tr>
<td>HRESULT</td>
<td>EnableApiSecurityContext ([in] TApiSecurityContext Context)</td>
<td>This command enables an API security context for Internet Explorer scripts.</td>
<td></td>
</tr>
<tr>
<td>HRESULT</td>
<td>CreateSms ([in] TSmsMessageType MessageType,[in] BSTR TargetNumbers,[out, retval] ISmsMessage **pMessage)</td>
<td>This command creates an SmsMessage.</td>
<td></td>
</tr>
<tr>
<td>HRESULT</td>
<td>SendSms ([in] BSTR TargetNumbers,[in] BSTR MessageText,[in, defaultvalue(&quot;&quot;)) BSTR ReplyToNumber, [out, retval] ISmsMessage **pMessage)</td>
<td>This command sends an SMS message.</td>
<td></td>
</tr>
<tr>
<td>HRESULT</td>
<td>AsyncSearchUsers ([in] BSTR Target,[out, retval] LONG *pCookie)</td>
<td>This command can be used to search for Skype users.</td>
<td></td>
</tr>
<tr>
<td>HRESULT</td>
<td>CreateChatUsingBlob ([in] BSTR Blob,[out, retval] IChat **pChat)</td>
<td>Returns existing or joins a new chat using given blob.</td>
<td></td>
</tr>
</tbody>
</table>
## Properties

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LONG</td>
<td><strong>Timeout</strong> [get, set]</td>
<td>This command queries/sets the wait timeout value in milliseconds.</td>
</tr>
<tr>
<td>BSTR</td>
<td><strong>Property</strong> ([in] BSTR ObjectType,[in] BSTR ObjectID,[in] BSTR PropName) [get, set]</td>
<td>This command queries/sets the properties of an object.</td>
</tr>
<tr>
<td>BSTR</td>
<td><strong>Variable</strong> ([in] BSTR Name) [get, set]</td>
<td>This command queries/sets Skype general parameters.</td>
</tr>
<tr>
<td>BSTR</td>
<td><strong>CurrentUserHandle</strong> [get]</td>
<td>This command queries the Skype name of the current user.</td>
</tr>
<tr>
<td><strong>TUserStatus</strong></td>
<td><strong>CurrentUserStatus</strong> [get, set]</td>
<td>This command queries/sets the online status of the current user.</td>
</tr>
<tr>
<td><strong>TConnectionStatus</strong></td>
<td><strong>ConnectionStatus</strong> [get]</td>
<td>This command queries the connection status of the Skype client.</td>
</tr>
<tr>
<td><strong>VARIANT_BOOL</strong></td>
<td><strong>Mute</strong> [get, set]</td>
<td>This command queries/sets the mute status of the Skype client.</td>
</tr>
<tr>
<td>BSTR</td>
<td><strong>Version</strong> [get]</td>
<td>This command queries the application version of the Skype client.</td>
</tr>
<tr>
<td><strong>VARIANT_BOOL</strong></td>
<td><strong>Privilege</strong> ([in] BSTR Name) [get]</td>
<td>This command queries the Skype services (privileges) enabled for the Skype client.</td>
</tr>
<tr>
<td><strong>IUser</strong></td>
<td><strong>CurrentUser</strong> [get]</td>
<td>This command queries the current user object.</td>
</tr>
<tr>
<td>Interface</td>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>-----------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>IConversion</td>
<td>Convert</td>
<td>queries the conversion object.</td>
</tr>
<tr>
<td>IUserCollection</td>
<td>Friends</td>
<td>queries the users in a contact list.</td>
</tr>
<tr>
<td>ICallCollection</td>
<td>Calls</td>
<td>queries calls in call history.</td>
</tr>
<tr>
<td>ICallCollection</td>
<td>ActiveCalls</td>
<td>queries a list of active calls.</td>
</tr>
<tr>
<td>ICallCollection</td>
<td>MissedCalls</td>
<td>queries a list of missed calls.</td>
</tr>
<tr>
<td>IChatMessageCollection</td>
<td>Messages</td>
<td>queries chat messages which were sent/received by the user.</td>
</tr>
<tr>
<td>IChatMessageCollection</td>
<td>MissedMessages</td>
<td>queries a list of missed chat messages.</td>
</tr>
<tr>
<td>TAttachmentStatus</td>
<td>AttachmentStatus</td>
<td>queries the attachment status of the Skype client.</td>
</tr>
<tr>
<td>LONG</td>
<td>Protocol</td>
<td>queries/sets the protocol version used by the Skype client.</td>
</tr>
<tr>
<td>IUser</td>
<td>User</td>
<td>queries a user object.</td>
</tr>
<tr>
<td>IChatMessage</td>
<td>Message</td>
<td>queries a chat message object.</td>
</tr>
<tr>
<td>Interface/Collection</td>
<td>Property/Method</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>----------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>ICall</td>
<td>Call (in, defaultvalue(0) LONG Id)</td>
<td>[get] This command queries a call object.</td>
</tr>
<tr>
<td>IChatCollection</td>
<td>Chats [get]</td>
<td>This command queries a list of chats.</td>
</tr>
<tr>
<td>IChat</td>
<td>Chat (in, defaultvalue(&quot;&quot;) BSTR Name)</td>
<td>[get] This command queries a chat object.</td>
</tr>
<tr>
<td>IConference</td>
<td>Conference (in, defaultvalue(0) LONG Id) [get]</td>
<td>This command queries a call conference object.</td>
</tr>
<tr>
<td>IConferenceCollection</td>
<td>Conferences [get]</td>
<td>This command queries a list of call conferences.</td>
</tr>
<tr>
<td>BSTR</td>
<td>Profile (in] BSTR Property) [get, set]</td>
<td>This command queries/sets user profile properties.</td>
</tr>
<tr>
<td>IChatCollection</td>
<td>ActiveChats [get]</td>
<td>This command queries a list of active chats.</td>
</tr>
<tr>
<td>IChatCollection</td>
<td>MissedChats [get]</td>
<td>This command queries a list of missed chats.</td>
</tr>
<tr>
<td>IChatCollection</td>
<td>RecentChats [get]</td>
<td>This command queries a list of recent chats.</td>
</tr>
<tr>
<td>IChatCollection</td>
<td>BookmarkedChats [get]</td>
<td>This command queries a list of bookmarked chats.</td>
</tr>
<tr>
<td>IVoicemailCollection</td>
<td>Voicemails [get]</td>
<td>This command queries a list of voicemails.</td>
</tr>
<tr>
<td>IUUserCollection</td>
<td>UsersWaitingAuthorization [get]</td>
<td>This command queries the list of users waiting for authorization.</td>
</tr>
<tr>
<td>Variant Type</td>
<td>Interface</td>
<td>Method</td>
</tr>
<tr>
<td>-------------</td>
<td>-----------</td>
<td>--------</td>
</tr>
<tr>
<td>VARIANT_BOOL</td>
<td>CommandId</td>
<td>[get, set]</td>
</tr>
<tr>
<td>IApplication</td>
<td>Application</td>
<td>([in] BSTR Name) [get]</td>
</tr>
<tr>
<td>IVoicemail</td>
<td>Greeting</td>
<td>([in, defaultvalue('&quot;&quot;')] BSTR Username) [get]</td>
</tr>
<tr>
<td>VARIANT_BOOL</td>
<td>Cache</td>
<td>[get, set]</td>
</tr>
<tr>
<td>IProfile</td>
<td>CurrentUserProfile</td>
<td>[get]</td>
</tr>
<tr>
<td>IGroupCollection</td>
<td>Groups</td>
<td>[get]</td>
</tr>
<tr>
<td>IGroupCollection</td>
<td>CustomGroups</td>
<td>[get]</td>
</tr>
<tr>
<td>IGroupCollection</td>
<td>HardwiredGroups</td>
<td>[get]</td>
</tr>
<tr>
<td>ISettings</td>
<td>Settings</td>
<td>[get]</td>
</tr>
<tr>
<td>IClient</td>
<td>Client</td>
<td>[get]</td>
</tr>
<tr>
<td>BSTR</td>
<td>FriendlyName</td>
<td>[set]</td>
</tr>
<tr>
<td>Command</td>
<td>([in] LONG Id,[in] BSTR</td>
<td></td>
</tr>
<tr>
<td>Interface</td>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------</td>
<td>-------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>ICommand</td>
<td>Command</td>
<td>This command sets the properties of a command object.</td>
</tr>
<tr>
<td></td>
<td>Reply</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Block</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Timeout (30000)</td>
<td></td>
</tr>
<tr>
<td>IVoicemail</td>
<td>Voicemail ([in] LONG Id)</td>
<td>This command queries the voicemail object.</td>
</tr>
<tr>
<td>IVoicemailCollection</td>
<td>MissedVoicemails [get]</td>
<td>This command requests a list of missed voicemails.</td>
</tr>
<tr>
<td>VARIANT_BOOL</td>
<td>ApiSecurityContextEnabled ([in] TApiSecurityContext Context) [get]</td>
<td>This command queries if an API security context for Internet Explorer is enabled.</td>
</tr>
<tr>
<td>ISmsMessageCollection</td>
<td>Smss [get]</td>
<td>This command requests a list of all SMS messages.</td>
</tr>
<tr>
<td>ISmsMessageCollection</td>
<td>MissedSmss [get]</td>
<td>This command requests a list of all missed SMS messages.</td>
</tr>
<tr>
<td>BSTR</td>
<td>ApiWrapperVersion [get]</td>
<td>This command returns API wrapper DLL version.</td>
</tr>
<tr>
<td>VARIANT_BOOL</td>
<td>SilentMode [get, set]</td>
<td>This command returns/sets Skype silent mode status.</td>
</tr>
<tr>
<td>IFileTransferCollection</td>
<td>FileTransfers [get]</td>
<td>Returns all file transfers.</td>
</tr>
<tr>
<td>IFileTransferCollection</td>
<td>ActiveFileTransfers [get]</td>
<td>Returns currently active file transfers.</td>
</tr>
<tr>
<td>IUserCollection</td>
<td>FocusedContacts [get]</td>
<td>Returns collection of selected contacts.</td>
</tr>
<tr>
<td>BSTR</td>
<td>PredictiveDialerCountry [get]</td>
<td></td>
</tr>
</tbody>
</table>
Returns predictive dialer country code.
**Detailed Description**

*Skype* is the default interface for the class.
Member Function Documentation

HRESULT AsyncSearchUsers ([in] BSTR Target, [out,retval] LONG * pCookie)

This command can be used to search for Skype users.

Parameters:
- [in] Target contains the person name or email address to be found.
- [out] pCookie receives search identifier.

See also: _ISkypeEvents::AsyncSearchUsersFinished

HRESULT Attach ([in, defaultvalue(5)] LONG Protocol, [in, defaultvalue(-1)] VARIANT_BOOL Wait)

This command establishes a connection to Skype.

Parameters:
- [in] Protocol is an optional field where you can enter the Skype protocol version, with a default value of 5.
- [in] Wait is an optional Boolean value which indicates whether or not to wait for a response before establishing a connection.

HRESULT ChangeUserStatus ([in] TUserStatus newVal)

This command changes the online status for the current user.
Parameters:

[in] **newVal** contains the new online status for the user.

Note:
This function waits until the online status changes. Alternatively, use the **ISkype::CurrentUserStatus** property to perform an immediate change of status.

**HRESULT ClearCallHistory ([in, defaultvalue("ALL")]) BSTR TCallHistory ([in, defaultvalue(chsAllCalls)])**

This command clears call history.

Parameters:

[in] **Username** is the Skypename of the user.

[in] **type** call type.

**HRESULT CreateChatMultiple ([in] IUserCollection * pMembers, [out, retval] IChat ** pVal)**

This command creates a chat with multiple users.

Parameters:

[in] **pMembers** contains an interface pointer to the **UserCollection** object.

[out] **pVal** receives the interface pointer for the **Chat** object.

See also:
**ISkype::CreateChatWith, IChat::AddMembers**

**HRESULT CreateChatUsingBlob ([in] BSTR Blob, **
Returns existing or joins a new chat using given blob.

**Parameters:**
- **[in]** Blob contains unique string indentifying the chat.
- **[out]** pChat receives the interface pointer for the Chat object.

```c
HRESULT CreateChatWith ([in] BSTR Username,
        [out, retval] IChat ** pVal
)
```

This command creates a chat with a single user.

**Parameters:**
- **[in]** Username contains the Skypename of the user.
- **[out]** pVal receives the interface pointer for the Chat object.

See also: `ISkype::CreateChatMultiple, IChat::AddMembers`

```c
HRESULT CreateGroup ([in] BSTR GroupName,
        [out, retval] IGroup ** pGroup
)
```

This command creates a contact group.

**Parameters:**
- **[in]** GroupName group name.
- **[out]** pGroup contains the interface pointer for the Group object.
<table>
<thead>
<tr>
<th>Function Name</th>
<th>Description</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>HRESULT DeleteGroup ( [in] LONG GroupId )</td>
<td>This command deletes a contact group. Users in the contact group are moved to the All Contacts hardwired contact group.</td>
<td>Parameters: [in] GroupId group identifier.</td>
</tr>
<tr>
<td>HRESULT FindChatUsingBlob ( [in] BSTR Blob, [out retval] I Chat ** pChat )</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Returns existing chat using given blob.

**Parameters:**

- **Blob** [in] contains unique string indentifying the chat.
- **pChat** [out] receives the interface pointer for the Chat object.

```c
HRESULT PlaceCall ([in] BSTR Target,
                   [in, defaultvalue("")] BSTR Target2,
                   [in, defaultvalue("")] BSTR Target3,
                   [in, defaultvalue("")] BSTR Target4,
                   [out, retval] ICall ** pCall)
```

This command places a call to a single user or creates a conference call.

**Parameters:**

- **Target** [in] call target to place a single user call.
- **Target2** [in] is an optional field which can contain an additional participant for a conference call.
- **Target3** [in] is an optional field which can contain an additional participant for a conference call.
- **Target4** [in] is an optional field which can contain an additional participant for a conference call.
- **pCall** [out] receives the interface pointer for the Call object.

**Note:**

If multiple targets are specified, a conference call is created. The call target can be username, phone number, or speed dial code.
HRESULT ResetCache ( )

This command deletes all cache entries.

See also:
Refer to the Skype4COM Dev Notes for information about the API cache.

HRESULT SearchForUsers ( [in] BSTR Target, [out, retval] IUserCollection ** pVal )

This command searches for users.

Parameters:
  [in] Target search target.
  [out] pVal receives the interface pointer for the UserCollection object.

HRESULT SendCommand ( [in] ICommand * pCommand )

This command sends an API command.

Parameters:


This command sends a chat message.
### Parameters:

- **Username** ([in]) user handle is the Skypename of the user.
- **Text** ([in]) message text.
- **pMessage** ([out]) receives the interface pointer for the ChatMessage object.

---

### HRESULT SendSms ( [in] BSTR TargetNumbers,
in] BSTR MessageText,
in, defaultvalue("")) BSTR ReplyToNumber,
out, retval] ISmsMessage ** pMessage )

This command sends an SMS message.

### Parameters:

- **TargetNumbers** ([out]) contains a comma-separated list of target numbers.
- **MessageText** ([out]) contains text of the message to send.
- **ReplyToNumber** ([out]) contains the reply-to number.
- **pMessage** ([out]) receives the interface pointer for the SmsMessage object.

---

### HRESULT SendVoicemail ([in] BSTR Username,
out, retval] IVoicemail ** pVoicemail )

This command sends a voicemail to a specified user.

### Parameters:

- **Username** ([in]) contains the Skypename of the user.
- **pVoicemail** ([out]) receives the interface pointer for the Voicemail object.
### Property Documentation

**HRESULT ApiSecurityContextEnabled [get]**

This command queries if an API security context for Internet Explorer is enabled.

**Parameters:**
- [in] **Context** is the TApiSecurityContext value.

**HRESULT Application [get]**

This command queries an application object.

**Parameters:**
- [in] **Name** is the application name.

**HRESULT Cache [get, set]**

This command queries/sets the status of internal cache.

The internal API cache is used to cache Skype object properties and global parameters.

**HRESULT Call [get]**

This command queries a call object.

**Parameters:**
- [in] **Id** is an optional field for the call ID.

**HRESULT Calls [get]**
This command queries calls in call history.

**Parameters:**

- [in] **Target** optional call target.

**HRESULT Chat** [get]

This command queries a chat object.

**Parameters:**

- [in] **Name** is an optional field for the chat name.

**HRESULT Command** [get]

This command sets the properties of a command object.

**Parameters:**

- [in] **Id** is the command ID.
- [in] **Command** is the command string.
- [in] **Reply** is the expected reply string.
- [in] **Block** is a blocking indicator. If set to True, **ISkype::SendCommand** method waits for a response from Skype API.
- [in] **Timeout** is the blocking timeout in milliseconds.

**See also:**

**ISkype::SendCommand**

**HRESULT Conference** [get]

This command queries a call conference object.

**Parameters:**

- [in] **Id** is an optional field for the conference ID.
**HRESULT CustomGroups [get]**

This command queries the list of custom contact groups.

Custom groups are contact groups defined by the user.

**HRESULT Greeting [get]**

This command queries the greeting used as voicemail.

**Parameters:**
[**in**] **Username** is an optional field containing the Skypename of the user.

**HRESULT HardwiredGroups [get]**

This command queries the list of hardwired contact groups.

Hardwired groups are "smart" contact groups, defined by Skype, that cannot be removed.

**HRESULT Message [get]**

This command queries a chat message object.

**Parameters:**
[**in**] **Id** is an optional field for the message ID.

**HRESULT Messages [get]**

This command queries chat messages which were sent/received by the user.
Parameters:
  [in] **Target** message sender.

**HRESULT Mute** [get, set]

This command queries/sets the mute status of the Skype client.

**Note:**
This value can be set only when there is an active call.

**HRESULT PredictiveDialerCountry** [get]

Returns predictive dialer country code.

**Parameters:**
  [out] **pVal** receives the country code.

**HRESULT Privilege** [get]

This command queries the Skype services (privileges) enabled for the Skype client.

**Parameters:**
  [in] **Name** privilege name
  ("SKYPEOUT","SKYPEIN","VOICEMAIL").

**HRESULT Profile** [get, set]

This command queries/sets user profile properties.

**Parameters:**
  [in] **Property** property name (for example
  "PSTN_BALANCE", ...)
"PSTN_BALANCE_CURRENCY",
"FULLNAME", "BIRTHDAY", "SEX",
"LANGUAGES", "COUNTRY", "PROVINCE",
"CITY", "PHONE_HOME",
"PHONE_OFFICE", "PHONE_MOBILE",
"HOMEPAGE", "ABOUT")

HRRESULT Property [get, set]

This command queries/sets the properties of an object.

Parameters:

- [in] ObjectType object type
  ("USER","CALL","CHAT","CHATMESSAGE",""
- [in] ObjectId object id.
- [in] PropName property name.

Example:

```vbscript
Set oSkype = CreateObject("Skype4COM.Skype"
)  
oSkype.Attach
WScript.Echo oSkype.Property("USER","echo123","FULLNAME")
```

HRRESULT Timeout [get, set]

This command queries/sets the wait timeout value in milliseconds.

This timeout value applies to every command sent to the Skype API. If a response is not received during the timeout period, an error is returned.

HRRESULT User [get]
This command queries a user object.

**Parameters:**

[in] **Username** is an optional field for the Skypename of the user.

**HRESULT Variable** [get, set]

This command queries/sets Skype general parameters.

**Parameters:**

[in] **Name** variable name.

**Example:**

```vbscript
Set oSkype = CreateObject("Skype4COM.Skype")
oSkype.Attach
WScript.Echo oSkype.Variable("MUTE")
```

**HRESULT Voicemail** [get]

This command queries the voicemail object.

**Parameters:**

[in] **Id** id the voicemail id.

---

The documentation for this interface was generated from the following file:

- **Skype4COM.idl**

---

Copyright © 2006 Skype Limited. All rights reserved.
<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Properties
ISmsChunk Interface Reference

SmsChunk is the default interface for this class. More...

import "Skype4COM.idl";

List of all members.
### Properties

<table>
<thead>
<tr>
<th>Type</th>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LONG</td>
<td><strong>Id</strong> [get]</td>
<td>This command requests the ID of an SMS message chunk.</td>
</tr>
<tr>
<td>LONG</td>
<td><strong>CharactersLeft</strong> [get]</td>
<td>This command requests the number of characters left in an SMS message chunk.</td>
</tr>
<tr>
<td>BSTR</td>
<td><strong>Text</strong> [get]</td>
<td>This command queries the text in an SMS message chunk.</td>
</tr>
<tr>
<td>ISmsMessage</td>
<td><strong>Message</strong> [get]</td>
<td>This command queries the <strong>SmsMessage</strong> associated with a chunk.</td>
</tr>
</tbody>
</table>
Detailed Description

*SmsChunk* is the default interface for this class.

The documentation for this interface was generated from the following file:

- *Skype4COM.idl*

Copyright © 2006 Skype Limited. All rights reserved.
## Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Public Member Functions |
Properties
SmsChunkCollection is the default interface for this class. More...

import "Skype4COM.idl";

List of all members.
## Public Member Functions

<table>
<thead>
<tr>
<th>HRESULT</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HRESULT</td>
<td>Add ([in] ISmsChunk *pItem)</td>
<td>This command adds a chunk to an SmsChunkCollection.</td>
</tr>
<tr>
<td>HRESULT</td>
<td>Remove ([in] long Index)</td>
<td>This command removes a chunk from an SmsChunkCollection.</td>
</tr>
<tr>
<td>HRESULT</td>
<td>RemoveAll (void)</td>
<td>This command removes all chunks from an SmsChunkCollection.</td>
</tr>
</tbody>
</table>
## Properties

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LONG</td>
<td><strong>Count</strong></td>
<td>[get] This command queries the number of chunks in a <strong>SmsChunkCollection</strong>.</td>
</tr>
<tr>
<td>ISmsChunk</td>
<td><strong>Item</strong> ([in] LONG Index)</td>
<td>[get] This command requests a chunk with a specified index in an <strong>SmsChunkCollection</strong>.</td>
</tr>
<tr>
<td>IUnknown</td>
<td><strong>_NewEnum</strong></td>
<td>[get] This command requests the enumerator object of an <strong>SmsChunkCollection</strong>.</td>
</tr>
</tbody>
</table>
Detailed Description

*SmsChunkCollection* is the default interface for this class.
Member Function Documentation

HRESULT Add ([in] ISmsChunk * pItem)

This command adds a chunk to an SmsChunkCollection.

Parameters:
  [in] pItem contains the interface pointer to the SmsChunk object.

HRESULT Remove ([in] long Index)

This command removes a chunk from an SmsChunkCollection.

Parameters:
  [in] Index contains the index of the chunk to be remove from the SMS chunk collection.
Property Documentation

HRESULT Item [get]

This command requests a chunk with a specified index in an SmsChunkCollection.

Parameters:

[in] Index contains the index of the chunk to return.

The documentation for this interface was generated from the following file:

- Skype4COM.idl

Copyright © 2006 Skype Limited. All rights reserved.
# Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Files</strong></td>
<td><strong>Examples</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Public Member Functions | Properties
SmsMessage is the default interface for this class. More...

import "Skype4COM.idl";

List of all members.
## Public Member Functions

<table>
<thead>
<tr>
<th>HRESULT</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HRESULT</td>
<td><code>Send()</code></td>
<td>This command sends an SMS message.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><code>Delete()</code></td>
<td>This command deletes an SMS message.</td>
</tr>
</tbody>
</table>
### Properties

<table>
<thead>
<tr>
<th>Type</th>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LONG</td>
<td><strong>Id</strong> [get]</td>
<td>This command requests an SMS message ID.</td>
</tr>
<tr>
<td><strong>TSmsMessageType</strong></td>
<td><strong>Type</strong> [get]</td>
<td>This command queries an SMS message type.</td>
</tr>
<tr>
<td><strong>TSmsMessageStatus</strong></td>
<td><strong>Status</strong> [get]</td>
<td>This command queries the status of an SMS message.</td>
</tr>
<tr>
<td><strong>TSmsFailureReason</strong></td>
<td><strong>FailureReason</strong> [get]</td>
<td>This command queries the reason an SMS message failed.</td>
</tr>
<tr>
<td>VARIANT_BOOL</td>
<td><strong>IsFailedUnseen</strong> [get]</td>
<td>This command queries if a failed SMS message was unseen.</td>
</tr>
<tr>
<td>VARIANT_BOOL</td>
<td><strong>Seen</strong> [set]</td>
<td>This command sets an SMS message status read.</td>
</tr>
<tr>
<td>LONG</td>
<td><strong>Price</strong> [get]</td>
<td>This command queries the price of an SMS message.</td>
</tr>
<tr>
<td><strong>PricePrecision</strong> [get]</td>
<td>This command queries the price precision associated with an SMS message.</td>
<td></td>
</tr>
<tr>
<td>BSTR</td>
<td><strong>PriceCurrency</strong> [get]</td>
<td>This command queries price currency code for an SMS message.</td>
</tr>
<tr>
<td>BSTR</td>
<td><strong>ReplyToNumber</strong> [get, set]</td>
<td>This command queries/sets the reply-to number for an SMS message.</td>
</tr>
<tr>
<td><strong>ISmsTargetCollection</strong></td>
<td><strong>Targets</strong> [get]</td>
<td>This command queries the targets of an SMS message.</td>
</tr>
<tr>
<td>Type</td>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------</td>
<td>---------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>BSTR</td>
<td>Body [get, set]</td>
<td>This command queries/sets the text of an SMS message.</td>
</tr>
<tr>
<td>ISmsChunkCollection</td>
<td>Chunks [get]</td>
<td>This command queries the chunks in an SMS message.</td>
</tr>
<tr>
<td>DATE</td>
<td>Timestamp [get]</td>
<td>This command queries the timestamp of an SMS message.</td>
</tr>
<tr>
<td>BSTR</td>
<td>TargetNumbers [get, set]</td>
<td>This command queries/sets the targets numbers of an SMS message.</td>
</tr>
</tbody>
</table>
Detailed Description

*SmsMessage* is the default interface for this class.
Property Documentation

HRESULT Price [get]

This command queries the price of an SMS message.

See also:
   ISmsMessage::PricePrecision,
   ISmsMessage::PriceCurrency

HRESULT PriceCurrency [get]

This command queries price currency code for an SMS message.

See also:
   ISmsMessage::Price, ISmsMessage::PricePrecision

HRESULT PricePrecision [get]

This command queries the price precision associated with an SMS message.

See also:
   ISmsMessage::Price, ISmsMessage::PriceCurrency

The documentation for this interface was generated from the following file:

- Skype4COM.idl

Copyright © 2006 Skype Limited. All rights reserved.
Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td>Class List</td>
<td>Class Members</td>
</tr>
</tbody>
</table>

Public Member Functions | Properties
**ISmsMessageCollection Interface Reference**

*SmsMessageCollection* is the default interface for this class. More...

```plaintext
import "Skype4COM.idl";
```

List of all members.
## Public Member Functions

<table>
<thead>
<tr>
<th>HRESULT</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add ([in] ISmsMessage *plItem)</td>
<td>This command adds a new message to an SmsMessageCollection.</td>
<td></td>
</tr>
<tr>
<td>Remove ([in] long Index)</td>
<td>This command removes a message from an SmsMessageCollection.</td>
<td></td>
</tr>
<tr>
<td>RemoveAll (void)</td>
<td>This command removes all messages from an SmsMessageCollection.</td>
<td></td>
</tr>
</tbody>
</table>
## Properties

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LONG</td>
<td>Count</td>
<td>This command queries the number of messages in an <strong>SmsMessageCollection</strong>.</td>
</tr>
<tr>
<td>ISmsMessage</td>
<td>Item ([in] LONG Index)</td>
<td>This command requests a message with a specified index in an <strong>SmsMessageCollection</strong>.</td>
</tr>
<tr>
<td>IUnknown</td>
<td>_NewEnum</td>
<td>This command requests the enumerator object of an <strong>SmsMessageCollection</strong>.</td>
</tr>
</tbody>
</table>
Detailed Description

SmsMessageCollection is the default interface for this class.

See also:
   ISkype::Smss, ISkype::MissedSmss
**Member Function Documentation**

HRESULT Add ([in] ISmsMessage * pItem )

This command adds a new message to an SmsMessageCollection.

**Parameters:**

[in] pItem contains the interface pointer to an SmsMessage object.

HRESULT Remove ([in] long Index )

This command removes a message from an SmsMessageCollection.

**Parameters:**

[in] Index contains the index of the SMS message to be removed.
Property Documentation

HRESULT Item [get]

This command requests a message with a specified index in an SmsMessageCollection.

Parameters:
[in] Index contains the index of the message to be returned.

The documentation for this interface was generated from the following file:

- Skype4COM.idl

Copyright © 2006 Skype Limited. All rights reserved.
<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td>Class List</td>
<td>Class Members</td>
</tr>
</tbody>
</table>

Properties
ISmsTarget Interface Reference

**SmsTarget** is the default interface for this class. More...

```plaintext
import "Skype4COM.idl";

List of all members.
```
## Properties

<table>
<thead>
<tr>
<th>Class</th>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TSmsTargetStatus</td>
<td>Status</td>
<td>[get] This command queries the status of an SMS message target.</td>
</tr>
<tr>
<td>BSTR</td>
<td>Number</td>
<td>[get] This command queries the target number of an SMS message.</td>
</tr>
<tr>
<td>ISmsMessage</td>
<td>Message</td>
<td>[get] This command queries the SmsMessage related to the target.</td>
</tr>
</tbody>
</table>
Detailed Description

**SmsTarget** is the default interface for this class.

The documentation for this interface was generated from the following file:

- **Skype4COM.idl**

Copyright © 2006 Skype Limited. All rights reserved.
## Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Public Member Functions | Properties
ISmsTargetCollection Interface Reference

*SmsTargetCollection* is the default interface for this class. More...

import "Skype4COM.idl";

List of all members.
### Public Member Functions

<table>
<thead>
<tr>
<th>HRESULT</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HRESULT</td>
<td><strong>Add</strong> ([in] ISmsTarget *pItem)</td>
<td>This command adds a target to an <strong>SmsTargetCollection</strong>.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>Remove</strong> ([in] long Index)</td>
<td>This command removes a target from an <strong>SmsTargetCollection</strong>.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>RemoveAll</strong> (void)</td>
<td>This command removes all targets from an <strong>SmsTargetCollection</strong>.</td>
</tr>
</tbody>
</table>
## Properties

<table>
<thead>
<tr>
<th>Type</th>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LONG</td>
<td><strong>Count</strong> [get]</td>
<td>This command queries the number of targets in an <strong>SmsTargetCollection</strong>.</td>
</tr>
<tr>
<td>ISmsTarget</td>
<td><strong>Item</strong> ([in] LONG Index) [get]</td>
<td>This command queries a target with a specified index in an <strong>SmsTargetCollection</strong>.</td>
</tr>
<tr>
<td>IUnknown</td>
<td><strong>_NewEnum</strong> [get]</td>
<td>This command queries the enumerator object of an <strong>SmsTargetCollection</strong>.</td>
</tr>
</tbody>
</table>
Detailed Description

SmsTargetCollection is the default interface for this class.
Member Function Documentation

HRESULT Add ([in] ISmsTarget * pItem)

This command adds a target to an SmsTargetCollection.

Parameters:
   [in] pItem contains the interface pointer to the new SmsTarget object.

HRESULT Remove ([in] long Index)

This command removes a target from an SmsTargetCollection.

Parameters:
   [in] Index contains the index of the target to remove.
Property Documentation

HRESULT Item [get]

This command queries a target with a specified index in an SmsTargetCollection.

Parameters:

    [in] Index contains the index of the target to return.

The documentation for this interface was generated from the following file:

- Skype4COM.idl

Copyright © 2006 Skype Limited. All rights reserved.
# Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td>Class List</td>
<td>Class Members</td>
</tr>
</tbody>
</table>

Properties
User class default interface. This interface enables you to read user properties. More...

import "Skype4COM.idl";

List of all members.
### Properties

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BSTR</td>
<td><strong>Handle</strong></td>
<td>[get, set] This command queries/sets the Skypename of the user.</td>
</tr>
<tr>
<td>BSTR</td>
<td><strong>FullName</strong></td>
<td>[get] This command queries the full name of the user.</td>
</tr>
<tr>
<td>DATE</td>
<td><strong>Birthday</strong></td>
<td>[get] This command queries the birthday of the user.</td>
</tr>
<tr>
<td>TUserSex</td>
<td><strong>Sex</strong></td>
<td>[get] This command queries the sex of the user.</td>
</tr>
<tr>
<td>BSTR</td>
<td><strong>Country</strong></td>
<td>[get] This command queries the country of the user.</td>
</tr>
<tr>
<td>BSTR</td>
<td><strong>Province</strong></td>
<td>[get] This command queries the province of the user.</td>
</tr>
<tr>
<td>BSTR</td>
<td><strong>City</strong></td>
<td>[get] This command queries the city of the user.</td>
</tr>
<tr>
<td>BSTR</td>
<td><strong>PhoneHome</strong></td>
<td>[get] This command queries the home telephone number of the user.</td>
</tr>
<tr>
<td>BSTR</td>
<td><strong>PhoneOffice</strong></td>
<td>[get] This command queries the office telephone number of the user.</td>
</tr>
<tr>
<td>BSTR</td>
<td><strong>PhoneMobile</strong></td>
<td>[get] This command queries the mobile phone number of the user.</td>
</tr>
<tr>
<td>BSTR</td>
<td><strong>Homepage</strong></td>
<td>[get] This command queries the homepage url of the user.</td>
</tr>
<tr>
<td>BSTR</td>
<td><strong>About</strong></td>
<td>[get] This command queries the about text for the user.</td>
</tr>
<tr>
<td>VARIANT_BOOL</td>
<td><strong>HasCallEquipment</strong></td>
<td>[get] This command queries if a user has call equipment. In the current implementation, this</td>
</tr>
</tbody>
</table>
command always returns a value of true.

<table>
<thead>
<tr>
<th>Class</th>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TBuddyStatus</td>
<td>BuddyStatus [get, set]</td>
<td>This command queries/sets user buddy status.</td>
</tr>
<tr>
<td>VARIANT_BOOL</td>
<td>IsAuthorized [get, set]</td>
<td>This command queries/sets user authorization status.</td>
</tr>
<tr>
<td>VARIANT_BOOL</td>
<td>IsBlocked [get, set]</td>
<td>This command queries/sets whether a user is blocked or not.</td>
</tr>
<tr>
<td>BSTR</td>
<td>DisplayName [get, set]</td>
<td>This command queries/sets the display name for a user.</td>
</tr>
<tr>
<td>TOnlineStatus</td>
<td>OnlineStatus [get]</td>
<td>This command queries the online status of a user.</td>
</tr>
<tr>
<td>DATE</td>
<td>LastOnline [get]</td>
<td>This command queries the time when a user was last online.</td>
</tr>
<tr>
<td>BSTR</td>
<td>CountryCode [get]</td>
<td>This command queries the ISO country code of a user.</td>
</tr>
<tr>
<td>BSTR</td>
<td>ReceivedAuthRequest [get]</td>
<td>This command queries the authorization request text of a user.</td>
</tr>
<tr>
<td>BSTR</td>
<td>SpeedDial [get, set]</td>
<td>This command queries/sets the speed dial code of a user.</td>
</tr>
<tr>
<td>VARIANT_BOOL</td>
<td>CanLeaveVoicemail [get]</td>
<td>This command queries if it is possible to send voicemail to a user.</td>
</tr>
<tr>
<td>BSTR</td>
<td>MoodText [get]</td>
<td>This command queries the mood text of a user.</td>
</tr>
<tr>
<td>BSTR</td>
<td>Aliases [get]</td>
<td>This command queries the alias text of a user.</td>
</tr>
<tr>
<td>LONG</td>
<td>Timezone [get]</td>
<td>This command queries the timezone of a user.</td>
</tr>
<tr>
<td>Data Type</td>
<td>Property/Method</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>-------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>VARIANT_BOOL</td>
<td><code>IsCallForwardActive</code></td>
<td>[get] This command queries the call forwarding status of a user.</td>
</tr>
<tr>
<td>BSTR</td>
<td><code>Language</code></td>
<td>[get] This command queries the language of the user.</td>
</tr>
<tr>
<td>BSTR</td>
<td><code>LanguageCode</code></td>
<td>[get] This command queries the ISO language code of a user.</td>
</tr>
<tr>
<td>VARIANT_BOOL</td>
<td><code>IsVideoCapable</code></td>
<td>[get] This command queries if a user has video capability.</td>
</tr>
<tr>
<td>VARIANT_BOOL</td>
<td><code>IsSkypeOutContact</code></td>
<td>[get] This command queries whether a user is a SkypeOut contact.</td>
</tr>
<tr>
<td>LONG</td>
<td><code>NumberOfAuthBuddies</code></td>
<td>[get] This command queries the number of authenticated buddies of the contact.</td>
</tr>
<tr>
<td>BSTR</td>
<td><code>RichMoodText</code></td>
<td>[get] This property returns user rich mood text.</td>
</tr>
<tr>
<td>VARIANT_BOOL</td>
<td><code>IsVoicemailCapable</code></td>
<td>[get] This property returns user voicemail capability status.</td>
</tr>
</tbody>
</table>
Detailed Description

User class default interface. This interface enables you to read user properties.

The documentation for this interface was generated from the following file:

- Skype4COM.idl

Copyright 2006 Skype Limited. All rights reserved.
Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td>Class List</td>
<td>Class Members</td>
</tr>
</tbody>
</table>

Public Member Functions | Properties
UserCollection class default interface. More...

import "Skype4COM.idl";

List of all members.
## Public Member Functions

|HRESULT| **Add** ([in] **IUser** *pUser*)  
This command adds a new item to a **UserCollection**.|
|---|---|
|HRESULT| **Remove** ([in] long **Index**)  
This command removes an item from a **UserCollection**.|
|HRESULT| **RemoveAll** (void)  
This command removes all items from a **UserCollection**.|

## Properties

<table>
<thead>
<tr>
<th>Type</th>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LONG</td>
<td><strong>Count [get]</strong></td>
<td>This command queries the number of users in a <a href="#">UserCollection</a>.</td>
</tr>
<tr>
<td>IUnknown</td>
<td><em>NewEnum [get]</em></td>
<td>This command queries the enumerator object for a <a href="#">UserCollection</a>.</td>
</tr>
<tr>
<td>IUnknown</td>
<td><strong>Item ([in] LONG Index) [get]</strong></td>
<td>This command queries an item in a <a href="#">UserCollection</a> with a specified index value.</td>
</tr>
</tbody>
</table>
Detailed Description

`UserCollection` class default interface.
**Member Function Documentation**

**HRESULT Add ([in] IUser * pUser )**

This command adds a new item to a `UserCollection`.

**Parameters:**
- **[in] pUser** receives the interface pointer for the `User` object.

**HRESULT Remove ([in] long Index )**

This command removes an item from a `UserCollection`.

**Parameters:**
- **[in] Index** contains the item to be removed.
**Property Documentation**

**HRESULT Item [get]**

This command queries an item in a `UserCollection` with a specified index value.

**Parameters:**

[in] `Index` contains the item to be returned.

The documentation for this interface was generated from the following file:

- Skype4COM.idl

---

Copyright �� 2006 Skype Limited. All rights reserved.
## Classes

### Public Member Functions

### Properties
IVoicemail Interface Reference

Voicemail class default interface. More...

import "Skype4COM.idl";

List of all members.
# Public Member Functions

<table>
<thead>
<tr>
<th>HRESULT</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HRESULT</td>
<td><strong>Open ()</strong></td>
<td>This command opens and plays voicemail.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>StartPlayback ()</strong></td>
<td>This command starts playing downloaded voicemail.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>StopPlayback ()</strong></td>
<td>This command stops playing downloaded voicemail.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>Upload ()</strong></td>
<td>This command uploads recorded voicemail from a local computer to the voicemail server.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>Download ()</strong></td>
<td>This command downloads a voicemail object from the voicemail server to a local computer.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>StartRecording ()</strong></td>
<td>This command stops playing a voicemail greeting and starts recording a voicemail message.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>StopRecording ()</strong></td>
<td>This command ends the recording of a voicemail message.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>Delete ()</strong></td>
<td>This command deletes a voicemail object.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>StartPlaybackInCall ()</strong></td>
<td>This command starts playing downloaded voicemail during a call.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>SetUnplayed ()</strong></td>
<td>This command changes the status of a voicemail from played to unplayed.</td>
</tr>
</tbody>
</table>
**Properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TVoicemailType</td>
<td>Type</td>
<td>[get] This command queries the voicemail type.</td>
</tr>
<tr>
<td>BSTR PartnerHandle</td>
<td>BSTR</td>
<td>[get] This command queries the Skypename of the user receiving a voicemail.</td>
</tr>
<tr>
<td>BSTR PartnerDisplayName</td>
<td>BSTR</td>
<td>[get] This command queries the display name of the user receiving the voicemail.</td>
</tr>
<tr>
<td>TVoicemailStatus</td>
<td>Status</td>
<td>[get] This command queries the voicemail status.</td>
</tr>
<tr>
<td>TVoicemailFailureReason</td>
<td>FailureReason</td>
<td>[get] This command queries the reason a voicemail failed.</td>
</tr>
<tr>
<td>DATE Timestamp</td>
<td>DATE</td>
<td>[get] This command queries the timestamp of a voicemail.</td>
</tr>
<tr>
<td>LONG Duration</td>
<td>LONG</td>
<td>[get] This command queries the duration of a voicemail.</td>
</tr>
<tr>
<td>LONG AllowedDuration</td>
<td>LONG</td>
<td>[get] This command queries the permitted duration of a voicemail.</td>
</tr>
<tr>
<td>LONG Id</td>
<td>LONG</td>
<td>[get] This command queries a voicemail ID.</td>
</tr>
</tbody>
</table>
Detailed Description

Voicemail class default interface.

The documentation for this interface was generated from the following file:

- Skype4COM.idl

Copyright © 2006 Skype Limited. All rights reserved.
Skype4COM 1.0.36.0

Main Page | Related Pages | Modules | Classes
---|---|---|---
Files | Examples
Class List | Class Members

Public Member Functions | Properties
IVoicemailCollection Interface Reference

**VoicemailCollection** class default interface. More...

import "Skype4COM.idl";

List of all members.
## Public Member Functions

<table>
<thead>
<tr>
<th>HRESULT</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HRESULT</td>
<td><strong>Add</strong> ([in] IVoicemail *pItem)</td>
<td>This command adds a new item to a VoicemailCollection.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>Remove</strong> ([in] long Index)</td>
<td>This command removes an item from a VoicemailCollection.</td>
</tr>
<tr>
<td>HRESULT</td>
<td><strong>RemoveAll</strong> (void)</td>
<td>This command removes all items from a VoicemailCollection.</td>
</tr>
</tbody>
</table>
### Properties

<table>
<thead>
<tr>
<th>LONG</th>
<th>Count  [get]</th>
<th>This command returns the number of voicemails in a VoicemailCollection.</th>
</tr>
</thead>
<tbody>
<tr>
<td>IVoicemail</td>
<td>Item ([in] LONG Index) [get]</td>
<td>This command queries an item in a VoicemailCollection with a specified index.</td>
</tr>
<tr>
<td>IUnknown</td>
<td>_NewEnum [get]</td>
<td>This command queries a VoicemailCollection enumerator object.</td>
</tr>
</tbody>
</table>
Detailed Description

VoicemailCollection class default interface.

Use the VoicemailCollection to manage Voicemail objects.
## Member Function Documentation

**HRESULT Add ([in] IVoicemail * pItem)**

This command adds a new item to a **VoicemailCollection**.

**Parameters:**
- **[in] pItem** receives the interface pointer for the **Voicemail** object.

**HRESULT Remove ([in] long Index)**

This command removes an item from a **VoicemailCollection**.

**Parameters:**
- **[in] Index** contains the item to be removed.
Property Documentation

HRESULT Item [get]

This command queries an item in a VoicemailCollection with a specified index.

Parameters:
[\texttt{in}] \texttt{Index} contains the index of the item to be returned.

The documentation for this interface was generated from the following file:

- \texttt{Skype4COM.idl}

Copyright © 2006 Skype Limited. All rights reserved.
## Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| SKYPE4COMLib | Participant |          |         |
Participant Class Reference

This class manages participants in a conference call. More...

import "Skype4COM.idl";
Detailed Description

This class manages participants in a conference call.

See also:

IParticipant, IParticipantCollection, ICall::Participants

The documentation for this class was generated from the following file:

- Skype4COM.idl

Copyright Ⓒ 2006 Skype Limited. All rights reserved.
<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

SKYPE4COMLib  ParticipantCollection
ParticipantCollection Class Reference

This class manages collections of participants in a conference call.

More...

import "Skype4COM.idl";
Detailed Description

This class manages collections of participants in a conference call.

Use the ParticipantCollection class to manage collections of participants in conference calls. To query a collection class use For Each .. Next statements.

See also:
   IParticipant, IParticipantCollection, ICall::Participants

The documentation for this class was generated from the following file:

- Skype4COM.idl

Copyright © 2006 Skype Limited. All rights reserved.
<table>
<thead>
<tr>
<th>Files</th>
<th>Examples</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td>SKYPE4COMLib</td>
<td>PluginEvent</td>
</tr>
</tbody>
</table>
PluginEvent Class Reference

This class manages plug-in event. More...

import "Skype4COM.idl";
Detailed Description

This class manages plug-in event.

See also:

IPluginEvent, _ISkypeEvents::PluginEventClicked, IPluginMenuItem

The documentation for this class was generated from the following file:

- Skype4COM.idl
## Classes

### Files

- SKYPE4COMLib

### Examples

- PluginMenuItem
PluginMenuItem Class Reference

Plugin menu item. More...

import "Skype4COM.idl";
Detailed Description

Plugin menu item.

See also:

IPluginMenuItem, _ISkypeEvents::PluginMenuItemClicked, IPluginEvent

The documentation for this class was generated from the following file:

- Skype4COM.idl

Copyright © 2006 Skype Limited. All rights reserved.
## Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SKYPE4COMLib</th>
<th>Profile</th>
</tr>
</thead>
</table>
Profile Class Reference

This class manages the user profile. More...

import "Skype4COM.idl";
Detailed Description

This class manages the user profile.

See also:

IProfile

The documentation for this class was generated from the following file:

- Skype4COM.idl

Copyright Ⓡ 2006 Skype Limited. All rights reserved.
### Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SKYPE4COMLib</td>
<td>Settings</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
This class manages the **Skype** API general parameters.  

```java
import "Skype4COM.idl";
```
Detailed Description

This class manages the Skype API general parameters.

See also:
   ISettings

The documentation for this class was generated from the following file:

- Skype4COM.idl

Copyright © 2006 Skype Limited. All rights reserved.
# Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SKYPE4COMLib</td>
<td>Skype</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Skype Class Reference

Skype API COM wrapper. More...

import "Skype4COM.idl";
Detailed Description

**Skype API** COM wrapper.

**See also:**

*ISkype, _ISkypeEvents*

The documentation for this class was generated from the following file:

- **Skype4COM.idl**

Copyright © 2006 Skype Limited. All rights reserved.
## Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| SKYPE4COMLib | SmsChunk |
SmsChunk Class Reference

SMS message chunk. More...

import "Skype4COM.idl";
Detailed Description

SMS message chunk.

See also:
ISmsMessage, ISmsChunk, ISmsChunkCollection

The documentation for this class was generated from the following file:

- Skype4COM.idl

Copyright 2006 Skype Limited. All rights reserved.
Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td>Class List</td>
<td>Class Members</td>
</tr>
<tr>
<td>SKYPE4COMLib</td>
<td>SmsChunkCollection</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Collection class for **SmsChunk** objects. More...

import "Skype4COM.idl";
Detailed Description

Collection class for SmsChunk objects.

See also:

ISmsMessage, ISmsChunk, ISmsChunkCollection

The documentation for this class was generated from the following file:

- Skype4COM.idl

Copyright © 2006 Skype Limited. All rights reserved.
<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SKYPE4COMLib</td>
<td>SmsMessage</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
SmsMessage Class Reference

SMS message. More...

import "Skype4COM.idl";
Detailed Description

SMS message.

See also:

ISmsMessage, ISmsMessageCollection, ISmsTarget, ISmsChunk

The documentation for this class was generated from the following file:

- Skype4COM.idl

Copyright © 2006 Skype Limited. All rights reserved.
# Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Files</strong></td>
<td><strong>Examples</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| SKYPE4COMLib | SmsMessageCollection |
SmsMessageCollection Class Reference

Collection class for SMS messages. More...

import "Skype4COM.idl";
Detailed Description

Collection class for SMS messages.

See also:
   ISmsMessage, ISmsMessageCollection, ISmsTarget, ISmsChunk

The documentation for this class was generated from the following file:

- Skype4COM.idl

Copyright © 2006 Skype Limited. All rights reserved.
## Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td>Modules</td>
<td>Classes</td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td>SKYPE4COMLib</td>
<td>SmsTarget</td>
</tr>
</tbody>
</table>

**SKYPE4COMLib**

**SmsTarget**
SmsTarget Class Reference

SMS message target. More...

import "Skype4COM.idl";
Detailed Description

SMS message target.

See also:

ISmsMessage, ISmsTarget, ISmsTargetCollection

The documentation for this class was generated from the following file:

- Skype4COM.idl

Copyright © 2006 Skype Limited. All rights reserved.
<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

SKYPE4COMLib  SmsTargetCollection
SmsTargetCollection Class Reference

Collection class for SMS targets. More...

import "Skype4COM.idl";
Detailed Description

Collection class for SMS targets.

See also:

ISmsMessage, ISmsTarget, ISmsTargetCollection

The documentation for this class was generated from the following file:

- Skype4COM.idl

Copyright © 2006 Skype Limited. All rights reserved.
# Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td>Modules</td>
<td>Classes</td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td>Classes</td>
<td>Classes</td>
</tr>
</tbody>
</table>

**SKYPE4COMLib**

**User**
User Class Reference

This class manages user properties. More...

import "Skype4COM.idl";
Detailed Description

This class manages user properties.

The **User** class provides access to user properties.

**Example:**

```csharp
'// Get current user object
oUser = oSkype.CurrentUser

'// Get full name
oHandle = oUser.FullName

'// Get about text
oAbout = oUser.About
```

See also:

* IUser, IUserCollection, ISkype::Friends, ISkype::SearchForUsers, IChat::ActiveMembers, IChat::Adder, IChat::Members, IChat::Posters*

The documentation for this class was generated from the following file:

- **Skype4COM.idl**

---

Copyright © 2006 Skype Limited. All rights reserved.
UserCollection Class Reference

**User** collection. More...

import "Skype4COM.idl";
**Detailed Description**

**User** collection.

The **UserCollection** class returns **User** objects. To query items in a collection, use **For Each .. Next** statements.

**Example:**

```
'// To query contacts:
For Each oUser In oSkype.Friends
Next

'// To search for users:
For Each oUser In oSkype.Users("peeter")
Next
```

See also:

- **IUser**, **IUserCollection**, **ISkype::Friends**, **ISkype::SearchForUsers**, **IChat::ActiveMembers**, **IChat::Adder**, **IChat::Members**, **IChat::Posters**

The documentation for this class was generated from the following file:

- **Skype4COM.idl**

Copyright © 2006 Skype Limited. All rights reserved.
Voicemail Class Reference

This class manages Skype voicemail. More...

import "Skype4COM.idl";
Detailed Description

This class manages Skype voicemail.

See also: IVoicemail, IVoicemailCollection

The documentation for this class was generated from the following file:

- Skype4COM.idl

Copyright © 2006 Skype Limited. All rights reserved.
<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SKYPE4COMLib</td>
<td>VoicemailCollection</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
VoicemailCollection Class Reference

Voicemail collection. More...

import "Skype4COM.idl";
Detailed Description

Voicemail collection.

Use this class to return collections of Voicemail objects. To access items in a VoicemailCollection, use For Each .. Next statements.

See also: IVoicemail, IVoicemailCollection

The documentation for this class was generated from the following file:

- Skype4COM.idl

Copyright 2006 Skype Limited. All rights reserved.
Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| Class List | Class Members |
# Class Index

<table>
<thead>
<tr>
<th>A</th>
<th>C</th>
<th>G</th>
<th>I</th>
<th>P</th>
<th>S</th>
<th>U</th>
<th>V</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>A</strong></td>
<td><strong>C</strong></td>
<td><strong>G</strong></td>
<td><strong>I</strong></td>
<td><strong>P</strong></td>
<td><strong>S</strong></td>
<td><strong>U</strong></td>
<td><strong>V</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Application (SKYPE4COMLib)</th>
<th>Conference (SKYPE4COMLib)</th>
<th>IChatMember (SKYPE4COMLib)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ApplicationStream (SKYPE4COMLib)</td>
<td>ConferenceCollection (SKYPE4COMLib)</td>
<td>IChatMemberCollection (SKYPE4COMLib)</td>
</tr>
<tr>
<td><strong>ApplicationStreamCollection</strong> (SKYPE4COMLib)</td>
<td>Conversion (SKYPE4COMLib)</td>
<td><strong>I</strong></td>
</tr>
<tr>
<td><strong>A</strong></td>
<td><strong>G</strong></td>
<td><strong>I</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>A</strong></th>
<th><strong>C</strong></th>
<th><strong>G</strong></th>
<th><strong>I</strong></th>
<th><strong>P</strong></th>
<th><strong>S</strong></th>
<th><strong>U</strong></th>
<th><strong>V</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Call</strong> (SKYPE4COMLib)</td>
<td><strong>Group</strong> (SKYPE4COMLib)</td>
<td><strong>GroupCollection</strong> (SKYPE4COMLib)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>CallChannel</strong> (SKYPE4COMLib)</td>
<td><strong>IApplication</strong></td>
<td><strong>IConf</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>CallChannelCollection</strong> (SKYPE4COMLib)</td>
<td><strong>IApplicationStream</strong></td>
<td><strong>I</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>CallChannelManager</strong> (SKYPE4COMLib)</td>
<td><strong>IApplicationStreamCollection</strong></td>
<td><strong>ICall</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>CallChannelMessage</strong> (SKYPE4COMLib)</td>
<td><strong>ICallChannel</strong></td>
<td><strong>ICallChannelCollection</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>CallCollection</strong> (SKYPE4COMLib)</td>
<td><strong>ICallChannelManager</strong></td>
<td><strong>ICallChannelMessage</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Chat</strong> (SKYPE4COMLib)</td>
<td><strong>ICallChannelCollection</strong></td>
<td><strong>IGroup</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>ChatCollection</strong> (SKYPE4COMLib)</td>
<td><strong>ICallChannelManager</strong></td>
<td><strong>ICallChannelMessage</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>ChatMember</strong> (SKYPE4COMLib)</td>
<td><strong>ICallChannelManager</strong></td>
<td><strong>ICallChannelMessage</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>ChatMemberCollection</strong> (SKYPE4COMLib)</td>
<td><strong>ICallChannelMessage</strong></td>
<td><strong>IParticipant</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>C</td>
<td>G</td>
<td>I</td>
<td>P</td>
<td>S</td>
<td>U</td>
<td>V</td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
</tbody>
</table>

Copyright © 2006 Skype Limited. All rights reserved.
Here is a list of all documented class members with links to the class documentation for each member:

- `_NewEnum`: `IUserCollection`, `ICallChannelCollection`, `IFileTransferCollection`, `ISmsMessageCollection`, `ISmsTargetCollection`, `ISmsChunkCollection`, `IGroupCollection`, `IApplicationStreamCollection`, `IVoicemailCollection`, `IConferenceCollection`, `IChatCollection`, `IChatMemberCollection`, `IChatMessageCollection`, `ICallCollection`, `IParticipantCollection`
### Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>File List</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
File List

Here is a list of all documented files with brief descriptions:

Skype4COM.idl [code]

Copyright © 2006 Skype Limited. All rights reserved.
<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td><strong>Examples</strong></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Examples

Here is a list of all examples:

- Answer.vbs
- Attach.vbs
- Buttons.vbs
- Call.php
- Call.vbs
- Chats.vbs
- Chats2.vbs
- Client.vbs
- Command.vbs
- Conference.vbs
- Conferences.vbs
- FileTransfers.vbs
- Groupchat.vbs
- Groups.vbs
- Join.vbs
- MessageStatus.js
- MessageStatus.vbs
- MissedMessages.js
- MissedMessages.vbs
- Plugins.vbs
- Profile.vbs
- Response.vbs
- Search.vbs
- Settings.vbs
- Sms.vbs
- Smss.vbs
- VoicemailServer.cs
- VoicemailServer.vbs

Copyright  2006 Skype Limited. All rights reserved.
Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
This script sets up automatic answering of incoming calls.

Copyright © 2004-2006 Skype Limited. Licensed under BSD license.

```vbs
'// Create a Skype4COM object:
Set oSkype = WScript.CreateObject("Skype4COM.Skype","Skype_")

'// Start the Skype client:

'// Connect to the Skype API:
oSkype.Attach

'// Run an infinite loop to ensure the connection remains active:
Do While True
   WScript.Sleep(60000)
Loop

'// The AttachmentStatus event handler monitors attachment status and automatically attempts to reattach to the API following loss of connection:
Public Sub Skype_AttachmentStatus(ByVal aStatus)
   WScript.Echo "Attachment status " & oSkype.Convert.AttachmentStatusToText(aStatus)
   If aStatus = oSkype.Convert.TextToAttachmentStatus("AVAILABLE") Then oSkype.Attach() End If
End Sub

'// The CallStatus event handler monitors call status and if the status is "ringing" and it is an in
coming call, it attempts to answer the call:

Public Sub Skype_CallStatus(ByRef aCall, ByVal aStatus)
    WScript.Echo "Call " & aCall.Id & " status " & aStatus & " " & oSkype.Convert.CallStatusToText(aStatus)
    If oSkype.Convert.TextToCallStatus("RINGING") = aStatus And _
        (oSkype.Convert.TextToCallType("INCOMING_P2P") = aCall.Type Or _
        oSkype(Convert.TextToCallType("INCOMING_PSTN") = aCall.Type) Then
        WScript.Echo "Answering call from " & aCall.PartnerHandle
        If aCall.TargetIdentity <> "" Then
            WScript.Echo "Call target identity: " & aCall.TargetIdentity
        End If
        aCall.Answer()
    End If
End Sub
Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
This script listens for connections on the Skype API, connects when available, and holds the skypename of the user when a connection is successful.

Copyright © 2004-2006 Skype Limited. Licensed under BSD license.

```vbs
' // Create a Skype4COM object:
Set oSkype = WScript.CreateObject("Skype4COM.Skype ", "Skype_")

' // Start the Skype client:

' // Connect to the Skype API with protocol version 5 (the default version):
oSkype.Attach 5

' // When attached to the API, send repeated commands to the API to test if they succeed:
Do While True
    WScript.Sleep(1000)
    If oSkype.AttachmentStatus = oSkype.Convert.TextToAttachmentStatus("SUCCESS") Then
        WScript.StdOut.Write "PING..
        oSkype.SendCommand(oSkype.Command(0, "PING","PING",TRUE))
    End If
Loop
```
'// The AttachmentStatus event handler monitors attachment status and returns the Skypename of the user when the status is "success":
Public Sub Skype_AttachmentStatus(ByVal aStatus)
    WScript.Echo "Attachment status " & oSkype.Convert.AttachmentStatusToText(aStatus)
    If aStatus = oSkype.Convert.TextToAttachmentStatus("AVAILABLE") Then oSkype.Attach() End If
    If aStatus = oSkype.Convert.TextToAttachmentStatus("SUCCESS") Then WScript.Echo "Current user is " & oSkype.CurrentUserHandle End If
End Sub
<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Skype4COM 1.0.36.0
This script sends key press events to the Skype client.

Copyright © 2004-2006 Skype Limited. Licensed under BSD license.

```vbs
' // Create a Skype4COM object:
Set oSkype = WScript.CreateObject("Skype4COM.Skype","Skype_")

' // Start the Skype client:

' // Connect to the Skype API:
oSkype.Attach

' // Bring the Skype client into focus and open the Contacts tab:
oSkype.Client.Focus
oSkype.Client.OpenContactsTab()

' // Emulate typing "ECHO123" into the address bar of the Skype client (appears in the address bar or the Skype client):
oSkype.Client.ButtonPressed "E"
oSkype.Client.ButtonReleased "E"
WScript.Sleep(1000)
oSkype.Client.ButtonPressed "C"
oSkype.Client.ButtonReleased "C"
WScript.Sleep(1000)
oSkype.Client.ButtonPressed "H"
oSkype.Client.ButtonReleased "H"
WScript.Sleep(1000)
oSkype.Client.ButtonPressed "O"
```
oSkype.Client.ButtonReleased "0"
WScript.Sleep(1000)
oSkype.Client.ButtonPressed "1"
oSkype.Client.ButtonReleased "1"
WScript.Sleep(1000)
oSkype.Client.ButtonPressed "2"
oSkype.Client.ButtonReleased "2"
WScript.Sleep(1000)
oSkype.Client.ButtonPressed "3"
oSkype.Client.ButtonReleased "3"
WScript.Sleep(1000)

' // Emulate pressing the Green (YES) button and the client starts a call:
oSkype.Client.ButtonPressed "YES"
oSkype.Client.ButtonReleased "YES"
WScript.Sleep(10000)

' // Emulate pressing the Red (NO) button and the client ends the call:
oSkype.Client.ButtonPressed "NO"
oSkype.Client.ButtonReleased "NO"

' // The AttachmentStatus event handler monitors attachment status and automatically attempts to reattach to the API following loss of connection:
Public Sub Skype_AttachmentStatus(ByVal aStatus)
    WScript.Echo ">Attachment status " & oSkype.Convert.AttachmentStatusToText(aStatus)
    If aStatus = oSkype.Convert.TextToAttachmentStatus("AVAILABLE") Then oSkype.Attach() End If
End Sub

Copyright © 2006 Skype Limited. All rights reserved.
Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
This PHP script places a call through the Skype API, and monitors status while the call is in progress.

Copyright © 2004-2006 Skype Limited. Licensed under BSD license.

```php
<?php

// Event sink:
class _ISkypeEvents {

    function AttachmentStatus($status) {
        echo "Attachment status $status\n";
    }
    function CallStatus($call, $status) {
        echo "Call $call->id status $status\n";
    }
}

// Create a Skype4COM object:
$skype = new COM("Skype4COM.Skype");

// Create a sink object:
$sink =& new _ISkypeEvents();
$sink->convert = $skype->convert();

// Connect to the sink:
com_event_sink($skype, $sink, "_ISkypeEvents");

// Create a conversion object:
$convert = $skype->convert;
$convert->language = "en";

// Start the Skype client, minimized and with no s
plash screen:
if (!$skype->client()->isRunning()) {
    $skype->client()->start(true, true);
}

// If the user status is not "online", change user status to "online":
if ($skype->currentUserStatus() == $convert->textToUserStatus("OFFLINE")) {
    $skype->changeUserStatus($convert->textToUserStatus("ONLINE"));
}

// Create a user object
$user = $skype->user("echo123");

// Place a call
$call = $skype->PlaceCall($user->handle);

// Wait for the call to be "in progress" and report an error if:
while ($call->status <> $convert->textToCallStatus("INPROGRESS")) {
    if ($call->status == $convert->textToCallStatus("FAILED") ||
        $call->status == $convert->textToCallStatus("REFUSED") ||
        $call->status == $convert->textToCallStatus("CANCELLED") ||
        $call->status == $convert->textToCallStatus("FINISHED") ||
        $call->status == $convert->textToCallStatus("BUSY"))
        die ("Call status " . $convert->callStatusToTe
xt($call->status));
    else
        com_message_pump (500);
}

// Send dtmf tones:
com_message_pump (10000);
if ($call->status == $convert->textToCallStatus("INPROGRESS")) $call->dtmf = "0";
com_message_pump (500);
if ($call->status == $convert->textToCallStatus("INPROGRESS")) $call->dtmf = "1";
com_message_pump (500);
if ($call->status == $convert->textToCallStatus("INPROGRESS")) $call->dtmf = "2";
com_message_pump (500);
if ($call->status == $convert->textToCallStatus("INPROGRESS")) $call->dtmf = "3";
com_message_pump (500);
if ($call->status == $convert->textToCallStatus("INPROGRESS")) $call->dtmf = "4";
com_message_pump (500);
if ($call->status == $convert->textToCallStatus("INPROGRESS")) $call->dtmf = "5";
com_message_pump (500);
if ($call->status == $convert->textToCallStatus("INPROGRESS")) $call->dtmf = "6";
com_message_pump (500);
if ($call->status == $convert->textToCallStatus("INPROGRESS")) $call->dtmf = "7";
com_message_pump (500);
if ($call->status == $convert->textToCallStatus("INPROGRESS")) $call->dtmf = "8";
com_message_pump (500);
if ($call->status == $convert->textToCallStatus("INPROGRESS")) $call->dtmf = "9";
com_message_pump (500);
if ($call->status == $convert->textToCallStatus("INPROGRESS")) $call->dtmf = ";
com_message_pump (500);
if ($call->status == $convert->textToCallStatus("INPROGRESS")) $call->dtmf = ";

// Finish the call:
if ($call->status <> $convert->textToCallStatus("FINISHED")) $call->finish();

//Sleep:
com_message_pump (1000);
<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
This script places a call through the Skype API, and monitors status while the call is in progress.

Copyright © 2004-2006 Skype Limited. Licensed under BSD license.

'// Create a Skype4COM object:
Set oSkype = WScript.CreateObject("Skype4COM.Skype ", "Skype_")

'// Start the Skype client:

'// Declare the following Skype constants:
cUserStatus_Offline = oSkype.Convert.TextToUserStatus("OFFLINE")
cUserStatus_Online = oSkype.Convert.TextToUserStatus("ONLINE")
cCallStatus_Ringing = oSkype.Convert.TextToCallStatus("RINGING")
cCallStatus_Inprogress = oSkype.Convert.TextToCallStatus("INPROGRESS")
cCallStatus_Failed = oSkype.Convert.TextToCallStatus("FAILED")
cCallStatus_Refused = oSkype.Convert.TextToCallStatus("REFUSED")
cCallStatus_Cancelled = oSkype.Convert.TextToCallStatus("CANCELLED")
cCallStatus_Finished = oSkype.Convert.TextToCallStatus("FINISHED")
cCallStatus_Busy = oSkype.Convert.TextToCallStatus("BUSY")
cAttachmentStatus_Available = oSkype.Convert.TextToCallStatus("AVAILABLE")
oAttachmentStatus("AVAILABLE")

'// The PlaceCall command will fail if the user is offline. To avoid failure, check user status and change to online if necessary:
If cUserStatus_Offline = oSkype.CurrentUserStatus
Then oSkype.ChangeUserStatus(cUserStatus_Online) End If

'// Create a user object:
Set oUser = oSkype.User("echo123")
WScript.Echo "User " & oUser.Handle & " online status is " & oSkype.Convert.OnlineStatusToText(oUser.OnlineStatus)

'// Place a call:
Set oCall = oSkype.PlaceCall(oUser.Handle)

'// Wait until the call is in progress and return the relevant error if any other status occurs:
Do While oCall.Status <> cCallStatus_Inprogress
    If oCall.Status = cCallStatus_Failed Or _
        oCall.Status = cCallStatus_Refused Or _
        oCall.Status = cCallStatus_Cancelled Or _
        oCall.Status = cCallStatus_Finished Or _
        oCall.Status = cCallStatus_Busy Then
    End If
    WScript.Sleep(500)
Loop

'// Check sending DTMF tones for use with interactive voice response (IVR) applications:
WScript.Sleep(10000)
If cCallStatus_InProgress = oCall.Status Then oCal
1.DTMF = "0" End If
WScript.Sleep(500)
If cCallStatus_InProgress = oCall.Status Then oCall
1.DTMF = "1" End If
WScript.Sleep(500)
If cCallStatus_InProgress = oCall.Status Then oCall
1.DTMF = "2" End If
WScript.Sleep(500)
If cCallStatus_InProgress = oCall.Status Then oCall
1.DTMF = "3" End If
WScript.Sleep(500)
If cCallStatus_InProgress = oCall.Status Then oCall
1.DTMF = "4" End If
WScript.Sleep(500)
If cCallStatus_InProgress = oCall.Status Then oCall
1.DTMF = "5" End If
WScript.Sleep(500)
If cCallStatus_InProgress = oCall.Status Then oCall
1.DTMF = "6" End If
WScript.Sleep(500)
If cCallStatus_InProgress = oCall.Status Then oCall
1.DTMF = "7" End If
WScript.Sleep(500)
If cCallStatus_InProgress = oCall.Status Then oCall
1.DTMF = "8" End If
WScript.Sleep(500)
If cCallStatus_InProgress = oCall.Status Then oCall
1.DTMF = "9" End If
WScript.Sleep(500)
If cCallStatus_InProgress = oCall.Status Then oCall
1.DTMF = "#" End If
WScript.Sleep(500)
If cCallStatus_InProgress = oCall.Status Then oCall
1.DTMF = "*" End If
WScript.Sleep(500)

'// Finish the call:
If oCall.Status <> cCallStatus_Finished Then oCall.Finish() End If

'// The AttachmentStatus event handler monitors attachment status and automatically attempts to reattach to the API following loss of connection:
Public Sub Skype_AttachmentStatus(ByVal aStatus)
    WScript.Echo "Attachment status " & oSkype.Convert.AttachmentStatusToText(aStatus)
    If aStatus = cAttachmentStatus_Available Then oSkype.Attach() End If
End Sub

'// If the call status is "failed", the CallStatus event handler returns the relevant failure reason as text:
Public Sub Skype_CallStatus(ByRef aCall, ByVal aStatus)
    WScript.Echo "Call " & aCall.Id & " status " & aStatus & " " & oSkype.Convert.CallStatusToText(aStatus)
    If cCallStatus_Failed = aStatus Then
        WScript.Echo "Failure reason:" & oSkype.Convert.CallFailureReasonToText(aCall.FailureReason)
    End If
End Sub

Copyright © 2006 Skype Limited. All rights reserved.
Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
This script queries chats in chat history.

Copyright © 2004-2006 Skype Limited. Licensed under BSD license.

'// Create a Skype4COM object:
Set oSkype = WScript.CreateObject("Skype4COM.Skype","Skype_")

'// Start the Skype client:

'// Query the total number of messages in history:

'// Query the timestamp, name, and display name of all chats in history:
WScript.Echo "All chats:"
For Each oChat In oSkype.Chats
    WScript.Echo oChat.Timestamp & " " & oChat.Name & " " & oChat.FriendlyName
Next

'// Query the timestamp, name, and display name of all active chats:
WScript.Echo "Active chats:"
For Each oChat In oSkype.ActiveChats
    WScript.Echo oChat.Timestamp & " " & oChat.Name & " " & oChat.FriendlyName
Next

'// Query the timestamp, name, and display name of
all missed chats:
WScript.Echo "Missed chats:"  
For Each oChat In oSkype.MissedChats  
  WScript.Echo oChat.Timestamp & " " & oChat.Name & " " & oChat.FriendlyName  
Next

'// Query the timestamp, name, and display name of all recent chats:
WScript.Echo "Recent chats:"  
For Each oChat In oSkype.RecentChats  
  WScript.Echo oChat.Timestamp & " " & oChat.Name & " " & oChat.FriendlyName  
Next

'// Query the timestamp, name, and display name of all bookmarked chats:
WScript.Echo "Bookmarked chats:"  
For Each oChat In oSkype.BookmarkedChats  
  WScript.Echo oChat.Timestamp & " " & oChat.Name & " " & oChat.FriendlyName  
Next

'// The AttachmentStatus event handler monitors attachment status and automatically attempts to reattach to the API following loss of connection:
Public Sub Skype_AttachmentStatus(ByVal aStatus)
  WScript.Echo "Attachment status " & oSkype.Convert.AttachmentStatusToText(aStatus)
  If aStatus = oSkype.Convert.TextToAttachmentStatus("AVAILABLE") Then oSkype.Attach() End If
End Sub

Copyright © 2006 Skype Limited. All rights reserved.
Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Chats2.vbs

An improved chats example to illustrate new chat properties.

Copyright © 2004-2006 Skype Limited. Licensed under BSD license.

'// Create Skype object
Set oSkype = CreateObject("Skype4COM.Skype")

'// Query properties of all recent chats:
WScript.Echo "Recent chats:"
For Each oChat In oSkype.RecentChats
    displayChatProperties(oChat)
Next

'// Query properties of all bookmarked chats:
WScript.Echo "Bookmarked chats:"
For Each oChat In oSkype.BookmarkedChats
    displayChatProperties(oChat)
Next

'// Displays chat properties
Function displayChatProperties(aChat)
    WScript.Echo vbCrLf & "Label: " & aChat.FriendlyName
    WScript.Echo "Created at: " & aChat.Timestamp
    WScript.Echo "Last activity at: " & aChat.ActivityTimestamp
    WScript.Echo "Id: " & aChat.Name
    WScript.Echo "Blob: " & aChat.Blob
    WScript.Echo "Description: " & aChat.Description
    WScript.Echo "My role: " & chatMemberRoleToText(aChat.MyRole)
    WScript.Echo "My status: " & chatStatusToText(aChat.MyStatus)
End Function
hat.MyStatus)
    WScript.Echo "Type: " & chatTypeToText(aChat.Type)
    If aChat.Type = chatTypeDialog Or aChat.Type = chatTypeLegacyDialog Then
        WScript.Echo "Dialog partner: " & aChat.DialogPartner
    End If
    If aChat.Members.Count > 0 Then
        WScript.StdOut.Write "Members:
        For Each oMember In aChat.Members
            WScript.StdOut.Write " " & oMember.Handle
        Next
        WScript.Echo ""
    End If
    If aChat.Applicants.Count > 0 Then
        WScript.StdOut.Write "Applicants:
        For Each oApplicant In aChat.Applicants
            WScript.StdOut.Write " " & oApplicant.Handle
        Next
        WScript.Echo ""
    End If
End Function

'// Converts chat types to plaintext
Function chatTypeToText(aType)
    Const chatTypeUnknown = -1
    Const chatTypeDialog = 0
    Const chatTypeLegacyDialog = 1
    Const chatTypeLegacyUnsubscribed = 2
    Const chatTypeMultiChat = 3
    Const chatTypeSharedGroup = 4
    Select Case aType
        Case chatTypeUnknown
            chatTypeToText = "Unknown"
        Case chatTypeDialog
            chatTypeToText = "Dialog"
Case chatTypeLegacyDialog
    chatTypeToText = "Legacy Dialog"
Case chatTypeLegacyUnsubscribed
    chatTypeToText = "Unsubscribed Legacy Dialog"
End Select
End Function

'// Converts chat roles to plaintext
Function chatMemberRoleToText(aRole)
    Const chatMemberRoleUnknown = -1
    Const chatMemberRoleCreator = 0
    Const chatMemberRoleMaster = 1
    Const chatMemberRoleHelper = 2
    Const chatMemberRoleUser = 3
    Const chatMemberRoleListener = 4
    Const chatMemberRoleApplicant = 5
    Select Case aRole
        Case chatMemberRoleUnknown
            chatMemberRoleToText = "Unknown"
        Case chatMemberRoleCreator
            chatMemberRoleToText = "Creator"
        Case chatMemberRoleMaster
            chatMemberRoleToText = "Master"
        Case chatMemberRoleHelper
            chatMemberRoleToText = "Helper"
        Case chatMemberRoleUser
            chatMemberRoleToText = "User"
        Case chatMemberRoleListener
            chatMemberRoleToText = "Listener"
        Case chatMemberRoleApplicant
            chatMemberRoleToText = "Applicant"
    End Select
End Function

'// Converts chat status to plaintext
Function chatStatusToText(aStatus)
    Const chatStatusUnknown = -1
    Const chatStatusConnecting = 0
    Const chatStatusWaitingRemoteAccept = 1
    Const chatStatusAcceptRequired = 2
    Const chatStatusPasswordRequired = 3
    Const chatStatusSubscribed = 4
    Const chatStatusUnsubscribed = 5
    Const chatStatusDisbanded = 6
    Const chatStatusQueuedBecauseChatIsFull = 7
    Const chatStatusApplicationDenied = 8
    Const chatStatusKicked = 9
    Const chatStatusBanned = 10
    Const chatStatusRetryConnecting = 11
    Select Case aStatus
        Case chatStatusUnknown
            chatStatusToText = "Unknown"
        Case chatStatusConnecting
            chatStatusToText = "Connecting"
        Case chatStatusWaitingRemoteAccept
            chatStatusToText = "Waiting for Remote Accept"
        Case chatStatusAcceptRequired
            chatStatusToText = "Accept Required"
        Case chatStatusPasswordRequired
            chatStatusToText = "Password Required"
        Case chatStatusSubscribed
            chatStatusToText = "Subscribed"
        Case chatStatusUnsubscribed
            chatStatusToText = "Unsubscribed"
        Case chatStatusDisbanded
            chatStatusToText = "Disbanded"
        Case chatStatusQueuedBecauseChatIsFull
            chatStatusToText = "Queued"
Case chatStatusApplicationDenied
    chatStatusToText = "Application Denied"

Case chatStatusKicked
    chatStatusToText = "Kicked"

Case chatStatusBanned
    chatStatusToText = "Banned"

Case chatStatusRetryConnecting
    chatStatusToText = "Reconnecting"

End Select
End Function
<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
This script opens different windows in the Skype client.

Copyright © 2004-2006 Skype Limited. Licensed under BSD license.

'//' Create a Skype4COM object:
Set oSkype = WScript.CreateObject("Skype4COM.Skype","Skype_")

'//' Start the Skype client:

'//' Bring the client into focus:
oSkype.Client.Focus
WScript.Sleep(1000)

'//' Minimize the client to the system tooltray:
oSkype.Client.Minimize
WScript.Sleep(1000)

'//' Open the Add Contacts window and enter "echo123" in the contact field:
oSkype.Client.OpenAddContactDialog "echo123"
WScript.Sleep(1000)

'//' Open a Chat window with "echo123" and type "Hi" in the message pane:
oSkype.Client.OpenMessageDialog "echo123", "Hi"
WScript.Sleep(1000)

'//' Transfer a file to "echo123"
oSkype.Client.OpenFileTransferDialog "echo123", "c:\temp"
'WScript.Sleep(1000)

'// Open other windows as required, with the following commands:
OSkype.Client.OpenProfileDialog
WScript.Sleep(1000)
OSkype.Client.OpenUserInfoDialog "echo123"
WScript.Sleep(1000)
OSkype.Client.OpenConferenceDialog
WScript.Sleep(1000)
OSkype.Client.OpenSearchDialog
WScript.Sleep(1000)
OSkype.Client.OpenOptionsDialog("GENERAL")
WScript.Sleep(1000)
OSkype.Client.OpenOptionsDialog("PRIVACY")
WScript.Sleep(1000)
OSkype.Client.OpenOptionsDialog("NOTIFICATIONS")
WScript.Sleep(1000)
OSkype.Client.OpenOptionsDialog("SOUNDALERTS")
WScript.Sleep(1000)
OSkype.Client.OpenOptionsDialog("SOUNDDEVICES")
WScript.Sleep(1000)
OSkype.Client.OpenOptionsDialog("HOTKEYS")
WScript.Sleep(1000)
OSkype.Client.OpenOptionsDialog("CONNECTION")
WScript.Sleep(1000)
OSkype.Client.OpenOptionsDialog("VOICEMAIL")
WScript.Sleep(1000)
OSkype.Client.OpenOptionsDialog("CALLFORWARD")
WScript.Sleep(1000)
OSkype.Client.OpenOptionsDialog("VIDEO")
WScript.Sleep(1000)
OSkype.Client.OpenOptionsDialog("ADVANCED")
WScript.Sleep(1000)
OSkype.Client.OpenCallHistoryTab
WScript.Sleep(1000)
OSkype.Client.OpenContactsTab
WScript.Sleep(1000)
oS?type.Client.OpenDialpadTab
WScript.Sleep(1000)
oS?type.Client.OpenLiveTab
WScript.Sleep(1000)
oS?type.Client.OpenSendContactsDialog  "echo123"
WScript.Sleep(1000)
oS?type.Client.OpenBlockedUsersDialog
WScript.Sleep(1000)
WScript.Sleep(1000)
WScript.Sleep(1000)
oS?type.Client.OpenAuthorizationDialog  "echo123"
WScript.Sleep(1000)
oS?type.Client.OpenVideoTestDialog
WScript.Sleep(1000)

'// The AttachmentStatus event handler monitors attachment status and automatically attempts to reattach to the API following loss of connection:
Public Sub Skype_AttachmentStatus(ByVal aStatus)
    WScript.Echo "Attachment status " & oSkype.Convert.AttachmentStatusToText(aStatus)
    If aStatus = oSkype.Convert.TextToAttachmentStatus("AVAILABLE") Then oSkype.Attach() End If
End Sub
# Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Command.vbs

This script sends blocking and non-blocking commands to the Skype API.

Copyright © 2004-2006 Skype Limited. Licensed under BSD license.

'// Create a Skype4COM object:
Set oSkype = WScript.CreateObject("Skype4COM.Skype","Skype_")

'// Start the Skype client:

'// Connect to the Skype API:
oSkype.Attach

'// Skype4COM can send two types of command; blocking commands and non-blocking commands. Blocking is defined as a Boolean variant of the command object and the default value is False, non-blocking. A blocking command awaits a reply before continuing. A non-blocking command does not require a reply.

'// Send the following non-blocking commands:
oSkype.SendCommand(oSkype.Command(0, "FOCUS"))
oSkype.SendCommand(oSkype.Command(1, "GET AUDIO_IN", "AUDIO_IN"))
oSkype.SendCommand(oSkype.Command(2, "GET AUDIO_OUT", "AUDIO_OUT"))
WScript.Sleep(1000)

'// Send the following blocking commands (note the
True (blocking) variant for each command):
oSsake.SendCommand(oSkype.Command(6, "GET USER echo123 COUNTRY", "USER echo123 COUNTRY", True))
WScript.Sleep(1000)

'//As with command objects, search commands can be defined as blocking or non-blocking, and the default value is False, non-blocking. A blocking search requires a response before activities continue. Following is a blocking search (note the True (blocking) variant:
Set oBlockingSearch = oSkype.Command(8888, "SEARCH USERS echo123", "USERS ", True)
oSsake.SendCommand(oBlockingSearch)
WScript.Echo "Search Result:" & oBlockingSearch.Reply

'//A non-blocking search does not require a response:
Set oSearchCommand = oSkype.Command(9999, "SEARCH USERS john doe", "USERS ")
oSsake.SendCommand(oSearchCommand)
WScript.Echo "Sleeping ..."
WScript.Sleep(30000)

'// The AttachmentStatus event handler monitors attachment status and automatically attempts to reattach to the API following loss of connection:
Public Sub Skype_AttachmentStatus(ByVal aStatus)
WScript.Echo "Attachment status " & oSkype.Convert.AttachmentStatusToText(aStatus)
    If aStatus = oSkype.Convert.TextToAttachmentStatus("AVAILABLE") Then oSkype.Attach() End If
End Sub

'// The reply event handler monitors the non-blocking search results:
Public Sub Skype_Reply(ByRef aCmd)
    If IsObject(oSearchCommand) Then
        If oSearchCommand.Id = aCmd.Id Then WScript.Echo "Search Finished!" End If
    End If
End Sub

Copyright © 2006 Skype Limited. All rights reserved.
Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Skype4COM 1.0.36.0
This script creates a conference call.

Copyright © 2004-2006 Skype Limited. Licensed under BSD license.

```vbs
'// Create a Skype4COM object:
Set oSkype = WScript.CreateObject("Skype4COM.Skype", "Skype_")

'// Start the Skype client:

'// The PlaceCall command will fail if the user is offline. To avoid failure, check user status and change to online if necessary:
If oSkype.Convert.TextToUserStatus("OFFLINE") = oSkype.CurrentUserStatus Then
End If

'// To create a conference call, use the PlaceCall command and add multiple users to the list of targets (to the maximum number of conference users supported by the OS environment):
Set oFirstCall = oSkype.PlaceCall("skypeuser1", "skypeuser2")
WScript.Sleep(30000)

'// The AttachmentStatus monitors and returns call status:
Public Sub Skype_CallStatus(ByRef aCall, ByVal aStatus)
```
WScript.Echo ">Call " & aCall.Id & " status " & aStatus & " " & oSkype.Convert.CallStatusToText(aStatus)
End Sub

'// The AttachmentStatus event handler monitors attachment status and automatically attempts to reattach to the API following loss of connection:
Public Sub Skype_AttachmentStatus(ByVal aStatus)
   WScript.Echo ">Attachment status " & oSkype.Convert.AttachmentStatusToText(aStatus)
   If aStatus = oSkype.Convert.TextToAttachmentStatus("AVAILABLE") Then oSkype.Attach() End If
End Sub

Copyright © 2006 Skype Limited. All rights reserved.
Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
This script lists conference calls hosted by a user, and lists participants in conference calls not hosted by a user.

Copyright © 2004-2006 Skype Limited. Licensed under BSD license.

'// Create a Skype4COM object:
Set oSkype = WScript.CreateObject("Skype4COM.Skype","Skype_")

'// Start the Skype client:

'// Count conference calls hosted by the current user. A conference call hosted by the current user is represented by multiple call objects which have a common conference ID (conf_id). For each conference object, return the conf_id, the number of calls, and the number of active calls:
For Each oConf In oSkype.Conferences
    WScript.Echo "Conference " & oConf.Id & " includes " & oConf.Calls.Count & " calls from which " & oConf.ActiveCalls.Count & " are active."
Next

'// To list conference calls in which the current user is active but not the host, begin by listing active calls for a user:
WScript.Echo "There are total " & oSkype.ActiveCalls.Count & " active calls."
Conference calls not hosted by the current user are all calls which have one or more participants (participants do not include the current user or the conference host). For each active call, return the Skypename and display name of each participant:

For Each oCall In oSkype.ActiveCalls

    If oCall.Participants.Count > 0 Then
        WScript.Echo "Conference call " & oCall.Id & " with " & oCall.PartnerHandle & " has " & oCall.Participants.Count & " participants."

        For Each oParticipant In oCall.Participants
            WScript.Echo "Participant " & oParticipant.Handle & " display name is " & oParticipant.DisplayName
        Next
    Else
        WScript.Echo "Active call " & oCall.Id & " with " & oCall.PartnerHandle & " is not a conference call."
    End If
Next

The AttachmentStatus event handler monitors attachment status and automatically attempts to reattach to the API following loss of connection:

Public Sub Skype_AttachmentStatus(ByVal aStatus)
    WScript.Echo "Attachment status " & oSkype.Convert.AttachmentStatusToText(aStatus)
    If aStatus = oSkype.Convert.TextToAttachmentStatus("AVAILABLE") Then oSkype.Attach() End If
End Sub

Copyright © 2006 Skype Limited. All rights reserved.
Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
FileTransfers.vbs

This script lists file transfers.

Copyright © 2004-2006 Skype Limited. Licensed under BSD license.

```vba
Set oSkype = WScript.CreateObject("Skype4COM.Skype", "Skype_"

For Each oTransfer In oSkype.FileTransfers
    FTDetails(oTransfer)
Next

For Each oTransfer In oSkype.ActiveFileTransfers
    FTDetails(oTransfer)
Next

WScript.Sleep(30000)

Public Sub Skype_FileTransferStatusChanged(ByRef aTransfer, ByVal aStatus)
    WScript.Echo ">File Transfer " & aTransfer.Id & " status " & FTStatusToText(aStatus)
End Sub

Sub FTDetails(aTransfer)
    WScript.Echo "Id:" & aTransfer.Id & " start:" & aTransfer.StartTime & " end:" & aTransfer.FinishTime
    WScript.Echo " Type:" & FTTypeToText(aTransfer.Type)
    WScript.Echo " Status:" & FTStatusToText(aTransfer.Status)
    WScript.Echo " Partner:" & aTransfer.PartnerHandle & "(" & aTransfer.PartnerDisplayName & ")"
```
WScript.Echo " File:" & aTransfer.FilePath
WScript.Echo " Bytes Transferred:" & aTransfer.BytesTransferred & " bytes"
WScript.Echo " Transfer Speed:" & aTransfer.BytesPerSecond & " bytes/sec"
WScript.Echo " Failure Reason:" & FTFailureReasonToText(aTransfer.FailureReason)
End Sub

Function FTTypeToText(aType)
    Const fileTransferTypeIncoming = 0
    Const fileTransferTypeOutgoing = 1

    Select Case aType
        Case fileTransferTypeIncoming
            FTTypeToText = "Incoming"
        Case fileTransferTypeOutgoing
            FTTypeToText = "Outgoing"
    End Select

End Function

Function FTStatusToText(aStatus)
    Const fileTransferStatusNew = 0
    Const fileTransferStatusConnecting = 1
    Const fileTransferStatusWaitingForAccept = 2
    Const fileTransferStatusTransferring = 3
    Const fileTransferStatusTransferringOverRelay = 4
    Const fileTransferStatusPaused = 5
    Const fileTransferStatusRemotelyPaused = 6
    Const fileTransferStatusCancelled = 7
    Const fileTransferStatusCompleted = 8
    Const fileTransferStatusFailed = 9

Select Case aStatus
  Case fileTransferStatusNew
    FTStatusToText = "New"
  Case fileTransferStatusConnecting
    FTStatusToText = "Connecting"
  Case fileTransferStatusWaitingForAccept
    FTStatusToText = "Waiting for Accept"
  Case fileTransferStatusTransferring
    FTStatusToText = "Transferring"
  Case fileTransferStatusTransferringOverRelay
    FTStatusToText = "Transferring over Relay"
  Case fileTransferStatusPaused
    FTStatusToText = "Paused"
  Case fileTransferStatusRemotelyPaused
    FTStatusToText = "Remotely Paused"
  Case fileTransferStatusCancelled
    FTStatusToText = "Cancelled"
  Case fileTransferStatusCompleted
    FTStatusToText = "Completed"
  Case fileTransferStatusFailed
    FTStatusToText = "Failed"
End Select

End Function

Function FTFailureReasonToText(aReason)
  Const fileTransferFailureReasonSenderNotAuthorized = 1
  Const fileTransferFailureReasonRemotelyCancelled = 2
  Const fileTransferFailureReasonFailedRead = 3
  Const fileTransferFailureReasonFailedRemoteRead = 4
  Const fileTransferFailureReasonFailedWrite = 5
  Const fileTransferFailureReasonFailedRemoteWrite = 6
End Function
Const fileTransferFailureReasonRemoteDoesNotSupportFT = 7
Const fileTransferFailureReasonRemoteOfflineTooLong = 8

Select Case aReason
    Case fileTransferFailureReasonSenderNotAuthorized
        FTFailureReasonToText = "Sender Not Authorized"
    Case fileTransferFailureReasonRemotelyCancelled
        FTFailureReasonToText = "Remotely Cancelled"
    Case fileTransferFailureReasonFailedRead
        FTFailureReasonToText = "Failed Read"
    Case fileTransferFailureReasonFailedRemoteRead
        FTFailureReasonToText = "Remote Read"
    Case fileTransferFailureReasonFailedWrite
        FTFailureReasonToText = "Write"
    Case fileTransferFailureReasonFailedRemoteWrite
        FTFailureReasonToText = "Remote Write"
    Case fileTransferFailureReasonRemoteDoesNotSupportFT
        FTFailureReasonToText = "Remote Does Not Support FT"
    Case fileTransferFailureReasonRemoteOfflineTooLong
        FTFailureReasonToText = "Remote Offline Too Long"
End Select
End Function

Copyright  © 2006 Skype Limited. All rights reserved.
Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
This script creates a group chat.

Copyright © 2004-2006 Skype Limited. Licensed under BSD license.

```vbs
'// Create a Skype4COM object:
Set oSkype = WScript.CreateObject("Skype4COM.Skype","Skype_")

'// Start the Skype client:

'// Create a UserCollection object and add the multichat users to this collection:
Set oMembers = CreateObject("Skype4COM.UserCollection")
oMembers.Add(oSkype.User("skypeuser2"))
oMembers.Add(oSkype.User("skypeuser3"))

'// Create a multichat:
Set oChat = oSkype.CreateChatMultiple(oMembers)

'// Open a chat window:
oChat.OpenWindow

'// Set a chat topic:
oChat.Topic = "Skype is cool"

'// Send a chat message:
Set oMsg = oChat.SendMessage("Greetings!")

'// Leave a chat:
oChat.Leave
```
'// The AttachmentStatus event handler monitors attachment status and automatically attempts to reattach to the API following loss of connection:
Public Sub Skype_AttachmentStatus(ByVal aStatus)
    WScript.Echo "Attachment status " & oSkype.Convert.AttachmentStatusToText(aStatus)
    If aStatus = oSkype.Convert.TextToAttachmentStatus("AVAILABLE") Then oSkype.Attach() End If
End Sub
<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Groups.vbs

This script manages various aspects of contact groups.

Copyright © 2004-2006 Skype Limited. Licensed under BSD license.

'// Create a Skype4COM object:
Set oSkype = WScript.CreateObject("Skype4COM.Skype", "Skype_")

'// Start the Skype client:

'// Create a custom group:
Set oMyGroup = oSkype.CreateGroup("MyGroup")

'// Add user "echo123" to the group:
oMyGroup.AddUser "echo123"

'// List the users in MyGroup and return the Skype name of each user:
For Each oUser In oMyGroup.Users
    WScript.Echo oUser.Handle & " (" & oUser.FullName & ")"
Next

'// Remove a user from a group:
oMyGroup.RemoveUser "echo123"

'// Delete a custom group:
oSkype.DeleteGroup oMyGroup.Id
'//' List all groups:
WScript.Echo "There are total " & oSkype.Groups.Count & " groups (" &

'//' List all custom groups:
WScript.Echo "Custom groups are:" & vbCrLf
For Each oGroup In oSkype.CustomGroups
  For Each oUser In oGroup.Users
    WScript.Echo oUser.Handle & " (" & oUser.FullName & ")"
  Next
  WScript.Echo ""
Next

'//' List all hardwired groups and return the Skype name and full name of each member of each group:
WScript.Echo "Hardwired groups are:" & vbCrLf
For Each oGroup In oSkype.HardwiredGroups
  For Each oUser In oGroup.Users
    WScript.Echo oUser.Handle & " (" & oUser.FullName & ")"
  Next
  WScript.Echo ""
Next

'//' Keep the script running for 60 seconds:
WScript.Sleep(60000)
'// The GroupVisible event handler returns information about whether a group is visible or hidden:
Public Sub Skype_GroupVisible(ByRef aGroup, ByVal aVisible)
    WScript.StdOut.Write "Group (" & aGroup.Id & ") type of [" & oSkype.Convert.GroupTypeToText(aGroup.Type) & "] labelled [" & aGroup.DisplayName & "]"
    If aVisible Then
        WScript.Echo " is visible."
    Else
        WScript.Echo " is hidden."
    End If
End Sub

'// The GroupExpanded event handler returns information about whether a group is expanded or collapsed:
Public Sub Skype_GroupExpanded(ByRef aGroup, ByVal aExpanded)
    WScript.StdOut.Write "Group (" & aGroup.Id & ") type of [" & oSkype.Convert.GroupTypeToText(aGroup.Type) & "] labelled [" & aGroup.DisplayName & "]"
    If aExpanded Then
        WScript.Echo " is expanded."
    Else
        WScript.Echo " is collapsed."
    End If
End Sub

'// The GroupUsers event handler gets the Skypenames of members of a group:
Public Sub Skype_GroupUsers(ByRef aGroup, ByRef aUsers)
    WScript.StdOut.Write "Group (" & aGroup.Id & ") type of [" & oSkype.Convert.GroupTypeToText(aGroup.Type) & "] labelled [" & aGroup.DisplayName & "] users"
    Dim oUser
For Each oUser In aUsers
    WScript.StdOut.Write " " & oUser.Handle
Next
WScript.Echo ""
End Sub

'// Delete a group:
Public Sub Skype_GroupDeleted(ByRef aGroupId)
    WScript.Echo "Group " & aGroupId & " deleted."
End Sub

'// Bring a contact into focus:
Public Sub Skype_ContactsFocused(ByVal aHandle)
    If Len(aHandle) Then
        WScript.Echo "Contact " & aHandle & " focused."
    Else
        WScript.Echo "No contact focused."
    End If
End If
End Sub

'// The AttachmentStatus event handler monitors attachment status and automatically attempts to reattach to the API following loss of connection:
Public Sub Skype_AttachmentStatus(ByVal aStatus)
    WScript.Echo "Attachment status " & oSkype.Convert.AttachmentStatusToText(aStatus)
    If aStatus = oSkype.Convert.TextToAttachmentStatus("AVAILABLE") Then oSkype.Attach() End If
End Sub

Copyright © 2006 Skype Limited. All rights reserved.
<table>
<thead>
<tr>
<th>Pages</th>
<th>Sections</th>
</tr>
</thead>
<tbody>
<tr>
<td>Main Page</td>
<td>Related Pages</td>
</tr>
<tr>
<td>Related Pages</td>
<td>Modules</td>
</tr>
<tr>
<td>Modules</td>
<td>Classes</td>
</tr>
<tr>
<td>Files</td>
<td>Examples</td>
</tr>
</tbody>
</table>
This script joins callers to form a conference call.

Copyright Ⓡ 2004-2006 Skype Limited. Licensed under BSD license.

'// Create a Skype4COM object:
Set oSkype = WScript.CreateObject("Skype4COM.Skype","Skype_")

'// Start the Skype client:

'// Declare the following Skype constants:
cUserStatus_Offline = oSkype.Convert.TextToUserStatus("OFFLINE")
cUserStatus_Online = oSkype.Convert.TextToUserStatus("ONLINE")
cCallStatus_Ringing = oSkype.Convert.TextToCallStatus("RINGING")
cCallStatus_Inprogress = oSkype.Convert.TextToCallStatus("INPROGRESS")
cCallStatus_Failed = oSkype.Convert.TextToCallStatus("FAILED")
cCallStatus_Refused = oSkype.Convert.TextToCallStatus("REFUSED")
cCallStatus_Cancelled = oSkype.Convert.TextToCallStatus("CANCELLED")
cCallStatus_Finished = oSkype.Convert.TextToCallStatus("FINISHED")
cCallStatus_Busy = oSkype.Convert.TextToCallStatus("BUSY")
cCallStatus_LocalHold = oSkype.Convert.TextToCallStatus("LOCALHOLD")
'// The PlaceCall command will fail if the user is offline. To avoid failure, check user status and change to online if necessary:
If cUserStatus_Offline = oSkype.CurrentUserStatus
Then oSkype.ChangeUserStatus(cUserStatus_Online) End If

'// Place a call (the first of more than one):
Set oCall1 = oSkype.PlaceCall("skypeuser1")
WaitForCallStatus oCall1, cCallStatus_Inprogress

'// Put the first call on hold:
oCall1.Hold
WaitForCallStatus oCall1, cCallStatus_LocalHold

'// Place the second call:
Set oCall2 = oSkype.PlaceCall("skypeuser2")
WaitForCallStatus oCall2, cCallStatus_Inprogress

'// Join these two calls to a conference:
WScript.Echo "Join call " & oCall2.Id & " to call " & oCall1.Id
oCall2.Join oCall1.Id

WScript.Sleep(60000)

'// The AttachmentStatus event handler monitors attachment status and automatically attempts to reattach to the API following loss of connection:
Public Sub Skype_AttachmentStatus(ByVal aStatus)
    WScript.Echo ">Attachment status " & oSkype.Convert.AttachmentStatusToText(aStatus)
    If aStatus = oSkype.Convert.TextToAttachmentStatus("AVAILABLE") Then oSkype.Attach() End If
End Sub
'// The CallStatus event handler monitors call status and returns status changes as text:
Public Sub Skype_CallStatus(ByVal oCall, ByVal Status)
    WScript.Echo "Call " & oCall.Id & " status " & Status & " " & oSkype.Convert.CallStatusToText(Status)
End Sub

'// Waits for call status to change:
Public Sub WaitForCallStatus(ByVal aCall, ByVal aStatus)
    Do While aCall.Status <> aStatus
        If aCall.Status = cCallStatus_Failed Or aCall.Status = cCallStatus_Refused Or aCall.Status = cCallStatus_Cancelled Or aCall.Status = cCallStatus_Finished Or aCall.Status = cCallStatus_Busy Then
            Err.Raise vbObjectError + 1, "", "Call " & aCall.Id & " " & oSkype.Convert.CallStatusToText(aCall.Status)
        End If
        WScript.Sleep(500)
    Loop
End Sub

Copyright © 2006 Skype Limited. All rights reserved.
Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
// Create a Skype4COM object:
var oSkype = new ActiveXObject("Skype4COM.Skype");

// Connect events:
WScript.ConnectObject(oSkype, "Skype_");

// Connect to Skype API:
oSkype.Attach();

// Loop and sleep:
while (true) {
    WScript.Sleep(1000);
}

// Chat message status events:
function Skype_MessageStatus(aMsg, aStatus) {
    if ((aStatus == oSkype.Convert.TextToChatMessageStatus("RECEIVED") ||
        aStatus == oSkype.Convert.TextToChatMessageStatus("SENT"))) &&
        aMsg.Type == oSkype.Convert.TextToChatMessageGetType("SAID")) {
        WScript.Echo(">Message " + aMsg.Id + " " + aMsg.FromHandle + " said " + aMsg.Body);
    }
}
Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
This script logs chat text messages.

Copyright © 2004-2006 Skype Limited. Licensed under BSD license.

'// Create a Skype4COM object:
Set oSkype = CreateObject("Skype4COM.Skype")

'// Connect events:
WScript.ConnectObject oSkype, "Skype_"

'// Connect to Skype API:
oSkype.Attach

'// Loop and sleep:
Do While True
    WScript.Sleep(60000)
Loop

'// Chat message status events:
Public Sub Skype_MessageStatus(ByRef aMsg, ByVal a Status)
    If (aStatus = oSkype.Convert.TextToChatMessageStatus("RECEIVED") Or _
        aStatus = oSkype.Convert.TextToChatMessageStatus("SENT") AndAlso _
        aMsg.Type = oSkype.Convert.TextToChatMessageType("SAID") Then
        WScript.Echo ">Message " & aMsg.Id & " " & aMsg.FromHandle & " said " & aMsg.Body
    End If
This script lists all missed messages and changes message status to "Read".

Copyright © 2004-2006 Skype Limited. Licensed under BSD license.

```javascript
// Create a Skype4COM object:
var oSkype = new ActiveXObject("Skype4COM.Skype");

// Query the total number of missed messages:
WScript.Echo("Total missed message count: " + oSkype.MissedMessages.Count);

// Query the id, timestamp, and sender Skypename of the message:
for (var i=0; i < oSkype.MissedMessages.Count; i++)
{
    var oMsg = oSkype.MissedMessages.Item(i+1);
    WScript.Echo("Msg id: " + oMsg.Id + " at " + oMsg.Timestamp + " from " + oMsg.FromHandle);
}

// Mark the missed message as "Read"
for (var i=0; i < oSkype.MissedMessages.Count; i++)
{
    var oMsg = oSkype.MissedMessages.Item(i+1);
    oMsg.Seen = true;
}
```

Copyright © 2006 Skype Limited. All rights reserved.
Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
MissedMessages.vbs

This script lists all missed messages and changes message status to "Read".

Copyright © 2004-2006 Skype Limited. Licensed under BSD license.

' // Create a Skype4COM object:
Set oSkype = WScript.CreateObject("Skype4COM.Skype","Skype_")

' // Query the total number of missed messages:

' // Query the id, timestamp, and sender Skypename of the message:
For Each oMsg In oSkype.MissedMessages
    WScript.Echo "Msg id: " & oMsg.Id & " at " & oMsg.Timestamp & " from " & oMsg.FromHandle
Next

' // Mark the missed message as "Read"
For Each oMsg In oSkype.MissedMessages
    oMsg.Seen = True
Next

Copyright © 2006 Skype Limited. All rights reserved.
Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
This script demos how to create plugin menu items and events.

Copyright © 2004-2006 Skype Limited. Licensed under BSD license.

```vbs
const pluginContextChat = 0
const pluginContextCall = 1
const pluginContextContact = 2
const pluginContextMyself = 3
const pluginContextTools = 4

const pluginContactTypeUndefined = -1
const pluginContactTypeAll = 0
const pluginContactTypeSkype = 1
const pluginContactTypeSkypeOut = 2

const pluginIconPath = "c:\temp\icon.png"

Set oSkype = WScript.CreateObject("Skype4COM.Skype", "Skype_"

Set oMenu1 = oSkype.Client.CreateMenuItem("menu1", pluginContextContact, "" &
   "Single Contact of Any Type", "Plugin that does cool stuff for single contact of any type", pluginIconPath, True, pluginContactTypeAll)

Set oMenu2 = oSkype.Client.CreateMenuItem("menu2", pluginContextContact, "" &
   "Single Skype Contact", "Plugin that does cool stuff for single Skype contact", pluginIconPath, True, pluginContactTypeSkype, False)

Set oMenu3 = oSkype.Client.CreateMenuItem("menu3", pluginIconPath, True)
```
Set oMenu4 = oSkype.Client.CreateMenuItem("menu4", pluginContextContact, "","&
   "Single SkypeOut Contact", "Plugin that does cool stuff for a single SkypeOut contact", pluginIconPath, True, pluginContactTypeSkypeOut, False)

Set oMenu5 = oSkype.Client.CreateMenuItem("menu5", pluginContextContact, "","&
   "Multiple SkypeOut Contacts", "Plugin that does cool stuff for a one or more SkypeOut contacts", pluginIconPath, True, pluginContactTypeSkypeOut, True)

Set oMenu6 = oSkype.Client.CreateMenuItem("menu6", pluginContextCall, "","&
   "Single Participant Call", "Plugin that does cool stuff in a call", pluginIconPath, True, pluginContactTypeUndefined, False)

Set oMenu7 = oSkype.Client.CreateMenuItem("menu7", pluginContextCall, "","&
   "Multiple Participants Call", "Plugin that does cool stuff in a conference call", pluginIconPath, True, pluginContactTypeUndefined, True)

Set oMenu8 = oSkype.Client.CreateMenuItem("menu8", pluginContextChat, "","&
   "Single Chat Member", "Plugin that does cool stuff for one-to-one chat", pluginIconPath, True, pluginContactTypeUndefined, False)

Set oMenu9 = oSkype.Client.CreateMenuItem("menu9", pluginContextChat, "","&
   "Multiple Chat Participants", "Plugin that does cool stuff in a chat", pluginIconPath, True, pluginContactTypeUndefined, True)
pluginContextChat, ""&_ "Multiple Chat Members", "Plugin that does cool stuff in group chat", pluginIconPath, True, plugin ContactTypeUndefined, True)

Set oMenu10 = oSkype.Client.CreateMenuItem("menu10", pluginContextTools, ""&_ "Tools plugin", "Plugin that does cool stuff in tools menu", pluginIconPath, True, pluginContactTypeUndefined, False)

Set oEvent = oSkype.Client.CreateEvent("event1", "Plugin example is running", "Click here to stop the example")
Do While Not bEventClicked
    WScript.Sleep(1000)
Loop
oEvent.Delete
oMenu1.Delete
oMenu2.Delete
oMenu3.Delete
oMenu4.Delete
oMenu5.Delete
oMenu6.Delete
oMenu7.Delete
oMenu8.Delete
oMenu9.Delete
oMenu10.Delete

Public Sub Skype_PluginEventClicked(ByVal aEvent)
    WScript.Echo ">Event " & aEvent.Id & " clicked."
    bEventClicked = True
End Sub

Public Sub Skype_PluginMenuItemClicked(ByRef aMenuItem, ByRef aUsers, ByVal aPluginContext, ByVal aContextId)
WScript.Echo ">Menu item ["& aMenuItem.Id &"] context [" & pluginContextToText(aPluginContext) & " ] id ["& aContextId &"] clicked."
    For Each oUser In aUsers
        WScript.Echo " User: "& oUser.FullName &" (" & oUser.Handle &")"
    Next
End Sub

Function pluginContextToText (context)
    Select Case context
    Case pluginContextChat
        pluginContextToText = "Chat"
    Case pluginContextCall
        pluginContextToText = "Call"
    Case pluginContextContact
        pluginContextToText = "Contact"
    Case pluginContextMyself
        pluginContextToText = "Myself"
    Case pluginContextTools
        pluginContextToText = "Tools"
    End Select
End Function

Copyright © 2006 Skype Limited. All rights reserved.
Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Profile.vbs

This script lists the profile properties for the current user.

Copyright © 2004-2006 Skype Limited. Licensed under BSD license.

'// Create a Skype4COM object:
Set oSkype = WScript.CreateObject("Skype4COM.Skype","Skype_")

'// Start the Skype client:

'// List the profile properties for the user:
WScript.Echo "Full name: " & oSkype.CurrentUserProfile.FullName
WScript.Echo "Languages: " & oSkype.CurrentUserProfile.Languages
WScript.Echo "Phone home: " & oSkype.CurrentUserProfile.PhoneHome
WScript.Echo "Phone mobile: " & oSkype.CurrentUserProfile.PhoneMobile
WScript.Echo "Call send to voicemail: " & oSkype.CurrentUserProfile.CallSendToVM

' // The AttachmentStatus event handler monitors attachment status and automatically attempts to reattach to the API following loss of connection:
Public Sub Skype_AttachmentStatus(ByVal aStatus)
    WScript.Echo "Attachment status: " & oSkype.Convert.AttachmentStatusToText(aStatus)
    If aStatus = oSkype.Convert.TextToAttachmentStatus("AVAILABLE") Then oSkype.Attach() End If
End Sub

Copyright © 2006 Skype Limited. All rights reserved.
Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
This script provides a response to a message received in a chat.

Copyright © 2004-2006 Skype Limited. Licensed under BSD license.

`'// Create a Skype4COM object:
Set oSkype = WScript.CreateObject("Skype4COM.Skype", "Skype_")

'// Start the Skype client:

'// Declare the following Skype constants:
cAttachmentStatus_Available = oSkype.Convert.TextToAttachmentStatus("AVAILABLE")
cMessageStatus_Sending = oSkype.Convert.TextToChatMessageStatus("SENDING")
cMessageStatus_Received = oSkype.Convert.TextToChatMessageStatus("RECEIVED")
cMessageType_Said = oSkype.Convert.TextToChatMessageType("SAID")
cMessageType_Left = oSkype.Convert.TextToChatMessageType("LEFT")

'// The SendMessage command will fail if the user is offline. To avoid failure, check user status and change to online if necessary:
If cUserStatus_Offline = oSkype.CurrentUserStatus Then oSkype.ChangeUserStatus(cUserStatus_Online) End If

'// Sleep
Do While True`
WScript.Sleep(60000)
Loop

'// The AttachmentStatus event handler monitors attachment status and attempts to connect to the Skype API:
Public Sub Skype_AttachmentStatus(ByVal aStatus)
    WScript.Echo ">Attachment status " & oSkype.Convert.AttachmentStatusToText(aStatus)
    If aStatus = cAttachmentStatus_Available Then oSkype.Attach() End If
End Sub

'// The MessageStatus event handler monitors message status, decodes received messages and, for those of type "Said", sends an autoresponse quoting the original message:
Public Sub Skype_MessageStatus(ByRef aMsg, ByVal aStatus)
    WScript.Echo ">Message " & aMsg.Id & " status " & oSkype.Convert.ChatMessageStatusToText(aStatus)
    If aStatus = cMessageStatus_Received Then
        DecodeMsg aMsg
        If aMsg.Type = cMessageType_Said Then
            'oSkype.SendMessage aMsg.FromHandle, "This is autoresponse."
            aMsg.Chat.SendMessage "You said [" & aMsg.Body & "]"
        End If
    End If
End Sub

'// The DecodeMsg event handler decodes messages in a chat and converts leave reasons to text for messages of type "Left":
Public Sub DecodeMsg(ByRef oMsg)
    sText = oMsg.FromHandle & " " & oSkype.Convert.C
hatMessageTypeToText(oMsg.Type) & ":"
    If len(oMsg.Body) Then
        sText = sText & " " & oMsg.Body
    End If
    Dim oUser
    For Each oUser In oMsg.Users
        sText = sText & " " & oUser.Handle
    Next
    If oMsg.Type = cMessageType_Left Then
        sText = sText & " " & oSkype.Convert.ChatLeaveReasonToText(oMsg.LeaveReason)
    End If
    WScript.Echo "->" & sText
End Sub
<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Skype4COM 1.0.36.0
Search.vbs

This script performs a simple search for a user.

Copyright © 2004-2006 Skype Limited. Licensed under BSD license.

'// Create a Skype4COM object:
Set oSkype = WScript.CreateObject("Skype4COM.Skype","Skype_")

'// Start the Skype client:

'// Search for users John Doe and return the Skypename, full name and country of each matching user:
Set oUsers = oSkype.SearchForUsers("john doe")
For Each oUser In oUsers
Next

'// The AttachmentStatus event handler monitors attachment status and automatically attempts to reattach to the API following loss of connection:
Public Sub Skype_AttachmentStatus(ByVal aStatus)
    WScript.Echo ">Attachment status " & oSkype.Convert.AttachmentStatusToText(aStatus)
    If aStatus = oSkype.Convert.TextToAttachmentStatus("AVAILABLE") Then oSkype.Attach() End If
End Sub

Copyright © 2006 Skype Limited. All rights reserved.
<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Settings.vbs

This script queries the Skype general parameters.

Copyright © 2004-2006 Skype Limited. Licensed under BSD license.

'// Create a Skype4COM object:
Set oSkype = WScript.CreateObject("Skype4COM.Skype","Skype_")

'// Start the Skype client:

'// Query Skype general parameters:
WScript.Echo "Audio in:" & oSkype.Settings.AudioIn
WScript.Echo "Ringer:" & oSkype.Settings.Ringer
'oSkype.Settings.Avatar = "c:\temp\test.jpg"
WScript.Echo "Ringtone status:" & oSkype.Settings.RingToneStatus
'oSkype.Settings.RingTone = "c:\temp\test.mp3"
WScript.Echo "Video in:" & oSkype.Settings.VideoIn
WScript.Echo "PC speaker:" & oSkype.Settings.PCSpeaker
WScript.Echo "AGC:" & oSkype.Settings.AGC
WScript.Echo "AEC:" & oSkype.Settings.AEC
'oSkype.Settings.Language = "en"
WScript.Echo "Language:" & oSkype.Settings.Language

'// The language change event
Public Sub Skype_UILanguageChanged(ByVal aCode)
    WScript.Echo "">Language changed to " & aCode
End Sub

'// The AttachmentStatus event handler monitors attachment status and automatically attempts to reattach to the API following loss of connection:
Public Sub Skype_AttachmentStatus(ByVal aStatus)
    WScript.Echo "">Attachment status " & oSkype.Convert.AttachmentStatusToText(aStatus)
    If aStatus = oSkype.Convert.TextToAttachmentStatus("AVAILABLE") Then oSkype.Attach() End If
End Sub

Copyright 2006 Skype Limited. All rights reserved.
<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Sms.vbs

This script sends a SMS message.

Copyright © 2004-2006 Skype Limited. Licensed under BSD license.

'// Create a Skype4COM object:
Set oSkype = WScript.CreateObject("Skype4COM.Skype", "Skype_")

'// Start the Skype client:

'// Send SMS:
Set oSMS = oSkype.SendSms("+1234567890", "Hello!")
WScript.Sleep(60000)

'// Message event handler:
Public Sub Skype_SmsMessageStatusChanged(ByRef aSms, ByVal aStatus)
    WScript.Echo ">Sms " & aSms.Id & " status " & aStatus & " " & oSkype(Convert.SmsMessageStatusToText(aStatus))
End Sub

'// Target event handler:
Public Sub Skype_SmsTargetStatusChanged(ByRef aTarget, ByVal aStatus)
    WScript.Echo ">Sms " & aTarget.Message.Id & " target " & aTarget.Number & " status " & aStatus & " " & oSkype(Convert.SmsTargetStatusToText(aStatus))
End Sub
### Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
This script lists all SMS messages.

Copyright © 2004-2006 Skype Limited. Licensed under BSD license.

```vbs
'// Create a Skype4COM object:
Set oSkype = WScript.CreateObject("Skype4COM.Skype", "Skype_")

'// Start the Skype client:

'// List Smss
For Each oSms In oSkype.Smss
    WScript.Echo "Sms Id: " & oSms.Id & " time " & oSms.Timestamp
    WScript.Echo "  type: " & oSms.Type & " " & oSkype.Convert.SmsMessageTypeToText(oSms.Type)
    WScript.Echo "  failure reason: " & oSms.FailureReason
    WScript.Echo "  failed unseen: " & oSms.IsFailedUnseen
    WScript.Echo "  price: " & oSms.Price
    WScript.Echo "  price precision: " & oSms.PricePrecision
    WScript.Echo "  price currency: " & oSms.PriceCurrency
    WScript.Echo "  reply to number: " & oSms.ReplyToNumber
    For Each oTarget In oSms.Targets
        WScript.Echo "  target: " & oTarget.Number & "
    Next oTarget
Next oSms
```
status: " & oSkype.Convert.SmsTargetStatusToText(oTarget.Status)
   Next
   WScript.Echo " body: [" & oSms.Body & "]"
   For Each oChunk In oSms.Chunks
      WScript.Echo " chunk: " & oChunk.Id & " [" & oChunk.Text & "]"
   Next
   Next

Copyright © 2006 Skype Limited. All rights reserved.
<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
<th>Files</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Skype4COM 1.0.36.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
VoicemailServer.cs

This script listens for new voicemails, sends SMS message and answers incoming calls.

Copyright © 2004-2006 Skype Limited. Licensed under BSD license.

using System;
using System.Collections.Generic;
using System.Text;
using SKYPE4COMLib;

namespace ConsoleApplication1
{
    class VoicemailServer
    {
        [MTAThread]
        static void Main(string[] args)
        {
            oSkype.CallStatus += Skype_CallStatus;
            oSkype.VoicemailStatus += Skype_VoicemailStatus;
            oSkype.Attach(5, true);
            while (true)
            {
                System.Threading.Thread.Sleep(1000);
            }
        }

        static void Skype_CallStatus(Call call, TC allStatus)
        {
            Console.WriteLine(">Call " + call.Id + " status: " + oConvert.CallStatusToText(status));
            if (status == TCallStatus.clsRinging &
                (call.Type == TCallType.cltIncomingP2P || call.T...
type == TCallType.cltIncomingPSTN)) {
        if (call.PartnerHandle == sMyMobileNumber) {
            call.Answer();
            foreach (Voicemail oVoicemail in oSkype.Voicemails) {
                if (oVoicemail.Status == TVoicemailStatus.vmsUnplayed) {
                    oVoicemail.StartPlaybackInCall();
                    while (oVoicemail.Status != TVoicemailStatus.vmsPlayed) {
                        System.Threading.Thread.Sleep(1000);
                    }
                    oVoicemail.SetUnplayed();
                }
            }
            if (call.Status == TCallStatus.clsInProgress) {
                call.Finish();
            }
        }
    }
    static void Skype_VoicemailStatus(Voicemail voicemail, TVoicemailStatus status) {
        Console.WriteLine(">Voicemail status: ");
        oConvert.VoicemailStatusToText(status);
        if (status == TVoicemailStatus.vmsDownloading) {
            Console.WriteLine("Sending SMS to ");
            Console.WriteLine(sMyMobileNumber);
SmsMessage oMessage = oSkype.SendMessage(sMyMobileNumber, "You have new voicemail from " + voicemail.PartnerDisplayName, "");

static Skype oSkype = new Skype();
static Conversion oConvert = new Conversion();
static String sMyMobileNumber = "+1234567890";
Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
VoicemailServer.vbs

This script listens for new voicemails, sends SMS message and answers incoming calls.

Copyright © 2004-2006 Skype Limited. Licensed under BSD license.

'// Create Skype object:
Set oSkype = WScript.CreateObject("Skype4COM.Skype", "Skype_")

'// Connect to Skype API:
oSkype.Attach

'// The mobile number used in this example
Const cMyMobileNumber = '+1234567890'

'// Skype constants
cCallStatusRinging = oSkype.Convert.TextToCallStatus("RINGING")
cCallStatusInprogress = oSkype.Convert.TextToCallStatus("INPROGRESS")
cCallTypeIncomingP2P = oSkype.Convert.TextToCallType("INCOMING_P2P")
cCallTypeIncomingPSTN = oSkype.Convert.TextToCallType("INCOMING_PSTN")
cVoicemailStatusUnplayed = oSkype.Convert.TextToVoicemailStatus("UNPLAYED")
cVoicemailStatusPlayed = oSkype.Convert.TextToVoicemailStatus("PLAYED")
cAttachmentStatus_Available = oSkype.Convert.TextToAttachmentStatus("AVAILABLE")

'// This script runs forever
  Do While True
WScript.Sleep(30000)
Loop

'// The CallStatus event handler monitors call status and if the status is "ringing" and it is an incoming call, answers the call:
Public Sub Skype_CallStatus(ByVal aCall, ByVal aStatus)
    WScript.Echo "Call " & aCall.Id & " status " & aStatus & " & oSkype.Convert.CallStatusToText(aStatus)
    If cCallStatusRinging = aStatus And _
        (cCallTypeIncomingP2P = aCall.Type Or cCallTypeIncomingPSTN = aCall.Type) Then

        '// Accept incoming call only from predefined phone number or Skypename:
        If aCall.PartnerHandle = cMyMobileNumber Then

            aCall.Answer()
            For Each oVoicemail In oSkype.Voicemails
                If oVoicemail.Status = cVoicemailStatusUnplayed Then
                    oVoicemail.StartPlaybackInCall
                    Do While oVoicemail.Status <> cVoicemailStatusPlayed
                        WScript.Sleep(10000)
                    Loop
                    '// Change the voicemail status back to "unplayed"
                    oVoicemail.SetUnplayed
                End If
            Next
        End If
    End If
End If
End Sub

'// The VoicemailStatus event handler monitors voicemail status changes:
Public Sub Skype_VoicemailStatus(aVoicemail, aStatus)
    WScript.Echo ">Voicemail " & aVoicemail.Id & " status " & oSkype.Convert.VoicemailStatusToText(aStatus)
    If cVoicemailStatusDownloading = aStatus Then
        WScript.Echo "Sending SMS to " & cMyMobileNumber
        SetoSms = oSkype.SendSms(cMyMobileNumber, "You have new voicemail from " & aVoicemail.PartnerDisplayName)
    End If
End Sub

'// The AttachmentStatus event handler monitors attachment status and automatically attempts to reattach to the API following loss of connection:
Public Sub Skype_AttachmentStatus(ByVal aStatus)
    WScript.Echo ">Attachment status " & oSkype.Convert.AttachmentStatusToText(aStatus)
    If aStatus = cAttachmentStatus_Available Then oSkype.Attach() End If
End Sub

Copyright © 2006 Skype Limited. All rights reserved.
Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Files</strong></td>
<td><strong>Examples</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>File List</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
// Skype4COM.idl : IDL source for Skype4COM.dll
import "oaidl.idl";
import "ocidl.idl";
#include "HelpString.h"
#include "olectl.h"

uuid(74A50AB8-1E6F-4246-8933-08E4FBBCF4E5),
helpstring(hsAttachStatus), helpcontext(17)
]
typedef enum TAttachmentStatus
{
    [helpcontext(17),helpstring(hsAttachStatus_Unknown)] apiAttachUnknown = -1,
    [helpcontext(17),helpstring(hsAttachStatus_Success)] apiAttachSuccess = 0,
    [helpcontext(17),helpstring(hsAttachStatus_Pending)] apiAttachPendingAuthorization = 1,
    [helpcontext(17),helpstring(hsAttachStatus_Refused)] apiAttachRefused = 2,
    [helpcontext(17),helpstring(hsAttachStatus_NotAvailable)] apiAttachNotAvailable = 3,
    [helpcontext(17),helpstring(hsAttachStatus_Available)] apiAttachAvailable = 4
} TAttachmentStatus;
typedef enum TConnectionStatus {
    [helpcontext(17), helpstring(hsConnStatus_Unknown)] conUnknown = -1,
    [helpcontext(17), helpstring(hsConnStatus_Offline)] conOffline = 0,
    [helpcontext(17), helpstring(hsConnStatus_Connecting)] conConnecting = 1,
    [helpcontext(17), helpstring(hsConnStatus_Pausing)] conPausing = 2,
    [helpcontext(17), helpstring(hsConnStatus_Online)] conOnline = 3
} TConnectionStatus;

typedef enum TUserStatus {
    [helpcontext(17), helpstring(hsUserStatus_Unknown)] cusUnknown = -1,
    [helpcontext(17), helpstring(hsUserStatus_Offline)] cusOffline = 0,
    [helpcontext(17), helpstring(hsUserStatus_Online)] cusOnline = 1,
    [helpcontext(17), helpstring(hsUserStatus_Away)] cusAway = 2,
    [helpcontext(17), helpstring(hsUserStatus_NotAvailable)] cusNotAvailable = 3,
} TUserStatus;
cusDoNotDisturb = 4,
cusInvisible = 5,
cusLoggedOut = 6,
cusSkypeMe = 7

} TUserStatus;

[ uuid(DCE03024-D231-45D4-AB69-D22C9E2954BA),
  helpstring(hsFailureReason), helpcontext(17) ]

typedef enum TCallFailureReason
{
  cfrUnknown = -1,
  cfrMiscError = 0,
  cfrUserDoesNotExist = 1,
  cfrUserIsOffline = 2,
  cfrNoProxyFound = 3,
  cfrSessionTerminated = 4,
  cfrNoCommonCodec = 5,
  cfrSoundIOError = 6,
  cfrRemoteDeviceError = 7,


enum TCallFailureReason {
    cfrBlockedByRecipient = 8,
    cfrRecipientNotFriend = 9,
    cfrNotAuthorizedByRecipient = 10,
    cfrSoundRecordingError = 11,
} TCallFailureReason;

typedef enum TCallStatus {
    clsUnknown = -1,
    clsUnplaced = 0,
    clsRouting = 1,
    clsEarlyMedia = 2,
    clsFailed = 3,
    clsRinging = 4,
    clsInProgress = 5,
    clsOnHold = 6,
    clsFinished = 7,
} TCallStatus;
us_Missed]) clsMissed = 8,
us_Refused]) clsRefused = 9,
us_BUSY] clsBusy = 10,
us_Cancelled]) clsCancelled = 11,
us_LocalHold]) clsLocalHold = 12,
us_RemoteHold]) clsRemoteHold = 13,
us_VmBuffGreeting]) clsVoicemailBufferingGreeting = 14,
us_VmPlayGreeting]) clsVoicemailPlayingGreeting = 15,
us_VmRecording]) clsVoicemailRecording = 16,
us_VmUploading]) clsVoicemailUploading = 17,
us_VmSent]) clsVoicemailSent = 18,
us_VmCancelled]) clsVoicemailCancelled = 19,
us_VmFailed]) clsVoicemailFailed = 20,
us_Transferring]) clsTransferring = 21,
us_Transferred]) clsTransferred = 22
} TCallStatus;

type TCallStatus

   [ uuid(38667A7C-DF57-4C1D-AC65-F7FC87B0E000),
     helpstring(hsCallType), helpcontext(17

typedef enum TCallType {
    [helpcontext(17), helpstring(hsCallType_Unknown)] cltUnknown = -1,
    [helpcontext(17), helpstring(hsCallType_IncomingPSTN)] cltIncomingPSTN = 0,
    [helpcontext(17), helpstring(hsCallType_OutgoingPSTN)] cltOutgoingPSTN = 1,
    [helpcontext(17), helpstring(hsCallType_IncomingP2P)] cltIncomingP2P = 2,
    [helpcontext(17), helpstring(hsCallType_OutgoingP2P)] cltOutgoingP2P = 3
} TCallType;

typedef enum TCallHistory {
    [helpcontext(17), helpstring(hsCallHistory_All)] chsAllCalls = 0,
    [helpcontext(17), helpstring(hsCallHistory_Missed)] chsMissedCalls = 1,
    [helpcontext(17), helpstring(hsCallHistory_Incoming)] chsIncomingCalls = 2,
    [helpcontext(17), helpstring(hsCallHistory_Outgoing)] chsOutgoingCalls = 3
} TCallHistory;
typedef enum TCallVideoStatus {
    [helpcontext(17), helpstring(hsCallVideoStatus_Unknown)] cvsUnknown = -1,
    [helpcontext(17), helpstring(hsCallVideoStatus_None)] cvsNone = 0,
    [helpcontext(17), helpstring(hsCallVideoStatus_SendEnabled)] cvsSendEnabled = 1,
    [helpcontext(17), helpstring(hsCallVideoStatus_RecvEnabled)] cvsReceiveEnabled = 2,
    [helpcontext(17), helpstring(hsCallVideoStatus_BothEnabled)] cvsBothEnabled = 3
} TCallVideoStatus;

typedef enum TCallVideoSendStatus {
    [helpcontext(17), helpstring(hsCallVideoSendStatus_Unknown)] vssUnknown = -1,
    [helpcontext(17), helpstring(hsCallVideoSendStatus_NotAvailable)] vssNotAvailable = 0,
    [helpcontext(17), helpstring(hsCallVideoSendStatus_Available)] vssAvailable = 1,
    [helpcontext(17), helpstring(hsCallVideoSendStatus_Starting)] vssStarting = 2,
    [helpcontext(17), helpstring(hsCallVideoSendStatus_Rejected)] vssRejected = 3,
    [helpcontext(17), helpstring(hsCallVideoSendStatus_Running)] vssRunning = 4,
vssStopping = 5,

vssPaused = 6

} TCallVideoSendStatus;

[uuid(8b0d63f5-57ec-4305-8f0d-517b6e85bad2),
 helpstring(hsCallIoDeviceType), helpcontext(17)]

typedef enum TCallIoDeviceType
{

callIoDeviceTypeUnknown = -1,
callIoDeviceTypeSoundcard = 0,
callIoDeviceTypePort = 1,
callIoDeviceTypeFile = 2
} TCallIoDeviceType;

[uuid(28B277BA-2D86-467D-BB47-A805A15585C1),
 helpstring(hsChatMessageType), helpcontext(17)]

typedef enum TChatMessageType
{

cmeUnknown = -1,
cmeCreatedChatWith = 0,
cmeCreatedChatWith
} TChatMessageType;
```c
    type_Saw]) cmeSawMembers = 1,
    [helpcontext(17), helpstring(hsChatMsgType_Saw)]
    type_Added]) cmeAddedMembers = 2,
    [helpcontext(17), helpstring(hsChatMsgType_Added)]
    type_Topic]) cmeSetTopic = 3,
    [helpcontext(17), helpstring(hsChatMsgType_Topic)]
    type_Said]) cmeSaid = 4,
    [helpcontext(17), helpstring(hsChatMsgType_Said)]
    type_Left]) cmeLeft = 5,
    [helpcontext(17), helpstring(hsChatMsgType_Left)]
    type_Emoted]) cmeEmoted = 6,
    [helpcontext(17), helpstring(hsChatMsgType_Emoted)]
    type_PostContacts]) cmePostedContacts = 7,
    [helpcontext(17), helpstring(hsChatMsgType_PostContacts)]
    type_GapInChat]) cmeGapInChat = 8,
    [helpcontext(17), helpstring(hsChatMsgType_GapInChat)]
    type_SetRole]) cmeSetRole = 9,
    [helpcontext(17), helpstring(hsChatMsgType_SetRole)]
    type_Kicked]) cmeKicked = 10,
    [helpcontext(17), helpstring(hsChatMsgType_Kicked)]
    type_SetOpts]) cmeSetOptions = 11,
    [helpcontext(17), helpstring(hsChatMsgType_SetOpts)]
    type_Kickbanned]) cmeKickBanned = 12,
    [helpcontext(17), helpstring(hsChatMsgType_Kickbanned)]
    type_Joined]) cmeJoinedAsApplicant = 13,
    [helpcontext(17), helpstring(hsChatMsgType_Joined)]
    type_SetPicture]) cmeSetPicture = 14,
    [helpcontext(17), helpstring(hsChatMsgType_SetPicture)]
    type_SetGuideline]) cmeSetGuidelines = 15,
    [helpcontext(17), helpstring(hsChatMessageStatus), helpcontext(17)]
} TChatMessageType;
```

typedef enum TChatMessageStatus {
    [helpcontext(17), helpstring(hsChatMsgStatus_Unknown)] cmsUnknown = -1,
    [helpcontext(17), helpstring(hsChatMsgStatus_Sending)] cmsSending = 0,
    [helpcontext(17), helpstring(hsChatMsgStatus_Sent)] cmsSent = 1,
    [helpcontext(17), helpstring(hsChatMsgStatus_Received)] cmsReceived = 2,
    [helpcontext(17), helpstring(hsChatMsgStatus_Read)] cmsRead = 3,
} TChatMessageStatus;

[uuid(AE9D03DE-7E88-466A-A3E8-1E840ACD9310),
 helpstring(hsChatMemberRole), helpcontext(17)]

typedef enum TChatMemberRole {
    [helpcontext(17), helpstring(hsChatMemberRoleUnknown)] chatMemberRoleUnknown = -1,
    [helpcontext(17), helpstring(hsChatMemberRoleCreator)] chatMemberRoleCreator = 0,
    [helpcontext(17), helpstring(hsChatMemberRoleMaster)] chatMemberRoleMaster = 1,
    [helpcontext(17), helpstring(hsChatMemberRoleHelper)] chatMemberRoleHelper = 2,
    [helpcontext(17), helpstring(hsChatMemberRoleUser)] chatMemberRoleUser = 3,
    [helpcontext(17), helpstring(""')] chatMemberRoleApplicant = 5,
} TChatMemberRole;
typedef enum TUserSex
{
    usexUnknown = -1,
    usexMale = 0,
    usexFemale = 1
} TUserSex;

typedef enum TBuddyStatus
{
    budUnknown = -1,
    budNeverBeenFriend = 0,
    budDeletedFriend = 1,
    budPendingAuthorization = 2,
    budFriend = 3
} TBuddyStatus;
typedef enum TOnlineStatus {
    [helpcontext(17),helpstring(hsOnlineStatus_Unknown)] olsUnknown = -1,
    [helpcontext(17),helpstring(hsOnlineStatus_Offline)] olsOffline = 0,
    [helpcontext(17),helpstring(hsOnlineStatus_Online)] olsOnline = 1,
    [helpcontext(17),helpstring(hsOnlineStatus_Away)] olsAway = 2,
    [helpcontext(17),helpstring(hsOnlineStatus_NotAvail)] olsNotAvailable = 3,
    [helpcontext(17),helpstring(hsOnlineStatus_DND)] olsDoNotDisturb = 4,
    [helpcontext(17),helpstring(hsOnlineStatus_SkypeOut)] olsSkypeOut = 5,
    [helpcontext(17),helpstring(hsOnlineStatus_SkypeMe)] olsSkypeMe = 6
} TOnlineStatus;

typedef enum TChatLeaveReason {
    [helpcontext(17),helpstring(hsLeaveReason_Unknown)] leaUnknown = -1,
    [helpcontext(17),helpstring(hsLeaveReason_NotFound)] leaUserNotFound = 0,
} TChatLeaveReason;
typedef enum TChatLeaveReason {
    leaUserIncapable = 1,
    leaAdderNotFriend = 2,
    leaAddedNotAuthorized = 3,
    leaAddDeclined = 4,
    leaUnsubscribe = 5
} TChatLeaveReason;

typedef enum TChatStatus {
    chsUnknown = -1,
    chsLegacyDialog = 0,
    chsDialog = 1,
    chsMultiNeedAccept = 2,
    chsMultiSubscribed = 3,
    chsUnsubscribed = 4
} TChatStatus;
typedef enum TChatType {
    [helpcontext(17), helpstring(hsChatTypeUnknown)] chatTypeUnknown = -1,
    [helpcontext(17), helpstring(hsChatTypeDialog)] chatTypeDialog = 0,
    [helpcontext(17), helpstring(hsChatTypeLegacy)] chatTypeLegacyDialog = 1,
    [helpcontext(17), helpstring(hsChatTypeLegacyUnsub)] chatTypeLegacyUnsubscribed = 2,
    [helpcontext(17), helpstring(hsChatTypeMultichat)] chatTypeMultiChat = 3,
    [helpcontext(17), helpstring(hsChatTypeSharedGrp)] chatTypeSharedGroup = 4,
} TChatType;

typedef enum TChatMyStatus {
    [helpcontext(17), helpstring(hsChatMyStatusUnknown)] chatStatusUnknown = -1,
    [helpcontext(17), helpstring(hsChatMyStatusConnecting)] chatStatusConnecting = 0,
    [helpcontext(17), helpstring(hsChatMyStatusWaitAccept)] chatStatusWaitingRemoteAccept = 1,
    [helpcontext(17), helpstring(hsChatMyStatusAcceptRequired)] chatStatusAcceptRequired = 2,
} TChatMyStatus;
[helpcontext(17),helpstring(hsChatMyStatus_PasswdRequired)] chatStatusPasswordRequired = 3,
[helpcontext(17),helpstring(hsChatMyStatus_Subscribed)] chatStatusSubscribed = 4,
[helpcontext(17),helpstring(hsChatMyStatus_Unsubscribed)] chatStatusUnsubscribed = 5,
[helpcontext(17),helpstring(hsChatMyStatus_Disbanded)] chatStatusDisbanded = 6,
[helpcontext(17),helpstring(hsChatMyStatus_Queued)] chatStatusQueuedBecauseChatIsFull = 7,
[helpcontext(17),helpstring(hsChatMyStatus_ApplicationDenied)] chatStatusApplicationDenied = 8,
[helpcontext(17),helpstring(hsChatMyStatus_Kicked)] chatStatusKicked = 9,
[helpcontext(17),helpstring(hsChatMyStatus_Banned)] chatStatusBanned = 10,
[helpcontext(17),helpstring(hsChatMyStatus_RetryConnecting)] chatStatusRetryConnecting = 11
,
} TChatMyStatus;

[uuid(371F92C4-78BC-45BB-B0B4-E9924C2B9EF9),
helpstring(""), helpcontext(17)
] typedef enum TChatOptions
{
[helpcontext(17),helpstring(hsChatOptions_JoinEnabled)] chatOptionJoiningEnabled = 1,
[helpcontext(17),helpstring(hsChatOptions_JoinerApplicant)] chatOptionJoinersBecomeApplicants = 2,
ons_JoinerListener)]]  chatOptionJoinersBecomeListeners = 4,
00343          [helpcontext(17),helpstring(hsChatOptions_HistoryDisclosed)]  chatOptionHistoryDisclosed = 8,
00344          [helpcontext(17),helpstring(hsChatOptions_UsersAreListeners)]  chatOptionUsersAreListeners = 16,
00345          [helpcontext(17),helpstring(hsChatOptions_TopicLocked)]  chatOptionTopicAndPictureLockedForUsers = 32,
00346          } TChatOptions;
00347
00348          [uuid(7771BE16-D5F0-4368-8A88-643B5B519A43),
00349          helpstring(hsVmType),  helpcontext(17)
00350          ]
00351          typedef enum TVoicemailType
00352          {
00353              [helpcontext(17),helpstring(hsVmType_U
00354              unknown)]  vmtUnknown = -1,
00355              [helpcontext(17),helpstring(hsVmType_I
00356              ncoming)]  vmtIncoming = 0,
00357              [helpcontext(17),helpstring(hsVmType_D
00358              efaultGreeting)]  vmtDefaultGreeting = 1,
00359              [helpcontext(17),helpstring(hsVmType_C
00360              ustomGreeting)]  vmtCustomGreeting = 2,
00361              [helpcontext(17),helpstring(hsVmType_O
00362              utgoing)]  vmtOutgoing = 3
00363          } TVoicemailType;
00364
00365          [uuid(F9810AFA-DBC3-4327-9876-E94A47E7C
00366          ED5),
00367          helpstring(hsVmStatus),  helpcontext(17)
typedef enum TVoicemailStatus {
  vmsUnknown = -1,
  vmsNotDownloaded = 0,
  vmsDownloading = 1,
  vmsUnplayed = 2,
  vmsPlaying = 4,
  vmsPlayed = 5,
  vmsBlank = 6,
  vmsRecording = 7,
  vmsRecorded = 8,
  vmsUploading = 9,
  vmsUploaded = 10,
  vmsDeleting = 11,
  vmsFailed = 12
} TVoicemailStatus;

[uuid(977E2F50-2361-4E81-B9F3-2303B8C8B49),
 helpstring(hsVmFailureReason), helpcon
typedef enum TVoicemailFailureReason
{
    [helpcontext(17),helpstring(hsVmFailureReason_Unknown)] vmrUnknown = -1,
    [helpcontext(17),helpstring(hsVmFailureReason_NoError)] vmrNoError = 0,
    [helpcontext(17),helpstring(hsVmFailureReason_MiscError)] vmrMiscError = 1,
    [helpcontext(17),helpstring(hsVmFailureReason_ConnectError)] vmrConnectError = 2,
    [helpcontext(17),helpstring(hsVmFailureReason_NoPrivilege)] vmrNoPrivilege = 3,
    [helpcontext(17),helpstring(hsVmFailureReason_NoVoicemail)] vmrNoVoicemail = 4,
    [helpcontext(17),helpstring(hsVmFailureReason_FileReadError)] vmrFileReadError = 5,
    [helpcontext(17),helpstring(hsVmFailureReason_FileWriteError)] vmrFileWriteError = 6,
    [helpcontext(17),helpstring(hsVmFailureReason_recordingError)] vmrRecordingError = 7,
    [helpcontext(17),helpstring(hsVmFailureReason_PlaybackError)] vmrPlaybackError = 8
} TVoicemailFailureReason;

typedef enum TGroupType
{
    [helpcontext(17),helpstring(hsGroupType_Unknown)] grpUnknown = -1,
} TGroupType;
e_Custom)] grpCustomGroup = 0,
00409  [helpcontext(17), helpstring(hsGroupType e_AllUsers)] grpAllUsers = 1,
00410  [helpcontext(17), helpstring(hsGroupType e_AllFriends)] grpAllFriends = 2,
00411  [helpcontext(17), helpstring(hsGroupType e_SkypeFriends)] grpSkypeFriends = 3,
00412  [helpcontext(17), helpstring(hsGroupType e_SkypeOutFriends)] grpSkypeOutFriends = 4,
00413  [helpcontext(17), helpstring(hsGroupType e_OnlineFriends)] grpOnlineFriends = 5,
00414  [helpcontext(17), helpstring(hsGroupType e_PendingAuthFriends)] grpPendingAuthorizationFriends = 6,
00415  [helpcontext(17), helpstring(hsGroupType e_RecentlyContactedUsers)] grpRecentlyContactedUsers = 7,
00416  [helpcontext(17), helpstring(hsGroupType e_WaitMyAuthorization)] grpUsersWaitingMyAuthorization = 8,
00417  [helpcontext(17), helpstring(hsGroupType e_AuthorizedByMe)] grpUsersAuthorizedByMe = 9,
00418  [helpcontext(17), helpstring(hsGroupType e_BlockedByMe)] grpUsersBlockedByMe = 10,
00419  [helpcontext(17), helpstring(hsGroupType e_UngroupedFriends)] grpUngroupedFriends = 11,
00420  [helpcontext(17), helpstring(hsGroupType e_SharedGroup)] grpSharedGroup = 12,
00421  [helpcontext(17), helpstring(hsGroupType e_ProposedSharedGroup)] grpProposedSharedGroup = 13
3
00422  } TGroupType;
00423
00424  [ uuid(46FF6EA6-E4B4-4E0A-B079-2E616F63396),
00425    helpstring(hsCallChannelType), helpcon
typedef enum TCallChannelType {
    [helpcontext(17), helpstring(hsCallChannelType_Unknown)] cctUnknown = -1,
    [helpcontext(17), helpstring(hsCallChannelType_Datagram)] cctDatagram = 0,
    [helpcontext(17), helpstring(hsCallChannelType_Reliable)] cctReliable = 1
} TCallChannelType;

typedef enum TApiSecurityContext {
    [helpcontext(17), helpstring(hsApiSecurityContext), helpcontext(17), hidden]
    [uuid(775c273f-6b14-4aa5-b33c-83127f41ba2e),]
    apiContextUnknown = 0x0000,
    [helpcontext(17), helpstring(hsApiSecurityContext_Voice)] apiContextVoice = 0x0001,
    [helpcontext(17), helpstring(hsApiSecurityContext_Messaging)] apiContextMessaging = 0x0002,
    [helpcontext(17), helpstring(hsApiSecurityContext_Account)] apiContextAccount = 0x0004,
    [helpcontext(17), helpstring(hsApiSecurityContext_Contacts)] apiContextContacts = 0x0008
} TApiSecurityContext;
typedef enum TSmsMessageType {
    smsMessageTypeUnknown = -1,
    smsMessageTypeIncoming = 0,
    smsMessageTypeOutgoing = 1,
    smsMessageTypeCCRequest = 2,
    smsMessageTypeCCSubmit = 3
} TSmsMessageType;

typedef enum TSmsMessageStatus {
    smsMessageStatusUnknown = -1,
    smsMessageStatusReceived = 0,
    smsMessageStatusRead = 1,
    smsMessageStatusComposing = 2,
    smsMessageStatusSendingToServer = 3,
} TSmsMessageStatus;
typedef enum TSmsMessageStatus {
    smsMessageStatusSentToServer = 4,
    smsMessageStatusDelivered = 5,
    smsMessageStatusSomeTargetsFailed = 6,
    smsMessageStatusFailed = 7
} TSmsMessageStatus;

typedef enum TSmsFailureReason {
    smsFailureReasonUnknown = -1,
    smsFailureReasonMiscError = 0,
    smsFailureReasonServerConnectFailed = 1,
    smsFailureReasonNoSmsCapability = 2,
    smsFailureReasonInsufficientFunds = 3,
    smsFailureReasonInvalidConfirmationCode = 4,
} TSmsFailureReason;
00490    [helpcontext(17),helpstring(hsSmsFailureReason_UserBlocked)] smsFailureReasonUserBlocked = 5,
00491    [helpcontext(17),helpstring(hsSmsFailureReason_IPBlocked)] smsFailureReasonIPBlocked = 6,
00492    [helpcontext(17),helpstring(hsSmsFailureReason_NodeBlocked)] smsFailureReasonNodeBlocked = 7
00493 } TSmsFailureReason;
00494
00495 [ uuid(34f2a71c-d453-4199-ab29-5b6e17b3fb5e),
00496    helpstring(hsSmsTargetStatus), helpcontext(17)
00497 ]
00498 typedef enum TSmsTargetStatus
00499 {
00500    [helpcontext(17),helpstring(hsSmsTargetStatus_Unknown)] smsTargetStatusUnknown = -1,
00501    [helpcontext(17),helpstring(hsSmsTargetStatus_Undefined)] smsTargetStatusUndefined = 0,
00502    [helpcontext(17),helpstring(hsSmsTargetStatus_Analyzing)] smsTargetStatusAnalyzing = 1,
00503    [helpcontext(17),helpstring(hsSmsTargetStatus_Acceptable)] smsTargetStatusAcceptable = 2,
00504    [helpcontext(17),helpstring(hsSmsTargetStatus_NotRoutable)] smsTargetStatusNotRoutable = 3,
00505    [helpcontext(17),helpstring(hsSmsTargetStatus_DeliveryPending)] smsTargetStatusDeliveryPending = 4,
00506    [helpcontext(17),helpstring(hsSmsTargetStatus_DeliverySuccessful)] smsTargetStatusDeliverySuccessful = 5,
typedef enum TSmsTargetStatus {
    smsTargetStatus_DeliveryFailed = 6,
} TSmsTargetStatus;

typedef enum TPluginContext {
    pluginContextUnknown = -1,
    pluginContextChat = 0,
    pluginContextCall = 1,
    pluginContextContact = 2,
    pluginContextMyself = 3,
    pluginContextTools = 4,
} TPluginContext;

typedef enum TPluginContactType {
    pluginContactTypeUnknown = -1,
} TPluginContactType;
typedef enum TPluginContactType {
    pluginContactTypeAll = 0,
    pluginContactTypeSkype = 1,
    pluginContactTypeSkypeOut = 2,
} TPluginContactType;

typedef enum TFileTransferType {
    fileTransferTypeIncoming = 0,
    fileTransferTypeOutgoing = 1,
} TFileTransferType;

typedef enum TFileTransferStatus {
    fileTransferStatusNew = 0,
} TFileTransferStatus;
sferStatus_Connecting) fileTransferStatusConnecting = 1,
fileTransferStatusWaitingForAccept = 2,
fileTransferStatusTransferring = 3,
fileTransferStatusTransferringOverRelay = 4,
fileTransferStatusPaused = 5,
fileTransferStatusRemotelyPaused = 6,
fileTransferStatusCancelled = 7,
fileTransferStatusCompleted = 8,
fileTransferStatusFailed = 9,
} TFileTransferStatus;

typedef enum TFileTransferFailureReason
{
    fileTransferFailureReason_SenderNotAuthorized = 1,
fileTransferFailureReasonRemotelyCancelled = 2,
fileTransferFailureReasonFailedRead = 3,
fileTransferFailureReasonFailedRemoteRead = 4,
fileTransferFailureReasonFailedWrite = 5,
fileTransferFailureReasonFailedRemoteWrite = 6,
fileTransferFailureReasonRemoteDoesNotSupportFT = 7,
fileTransferFailureReasonRemoteOfflineTooLong = 8,

} TFileTransferFailureReason;

[object,
uuid(48E046A8-31D7-4E5F-A611-47BF32B86405),
dual,
nonextensible,
pointer_default(unique),
oleautomation,
helpstring(hsICommand),
helpcontext(7)
]

interface ICommand : IDispatch
{
    [propget, id(1), helpstring(hsICommand_Id)],helpcontext(7)] HRESULT Id([out, retval] LONG
*pVal);
00595      [propput, id(1), helpstring(hsICommand
_Id),helpcontext(7)] HRESULT Id([in] LONG newVal);
00596      [propput, id(2), helpstring(hsICommand
_Timeout),helpcontext(7)] HRESULT Timeout([out, re
tval] LONG *pVal);
00597      [propput, id(2), helpstring(hsICommand
_Timeout),helpcontext(7)] HRESULT Timeout([in] LON
G newVal);
00598      [propput, id(3), helpstring(hsICommand
_Block),helpcontext(7)] HRESULT Blocking([out, ret
val] VARIANT_BOOL *pVal);
00599      [propput, id(3), helpstring(hsICommand
_Block),helpcontext(7)] HRESULT Blocking([in] VARI
ANT_BOOL newVal);
00600      [propput, id(4), helpstring(hsICommand
_Command),helpcontext(7)] HRESULT Command([out, re
tval] BSTR *pVal);
00601      [propput, id(4), helpstring(hsICommand
_Command),helpcontext(7)] HRESULT Command([in] BST
R newVal);
00602      [propput, id(5), helpstring(hsICommand
_Reply),helpcontext(7)] HRESULT Reply([out, retval
] BSTR *pVal);
00603      [propput, id(5), helpstring(hsICommand
_Reply),helpcontext(7)] HRESULT Reply([in] BSTR ne
wVal);
00604      [propput, id(6), helpstring(hsICommand
_Expect),helpcontext(7)] HRESULT Expected([out, ret
val] BSTR *pVal);
00605      [propput, id(6), helpstring(hsICommand
_Expect),helpcontext(7)] HRESULT Expected([in] BST
R newVal);
00606      ];
00607      [  
00608      object,
00609      uuid(D0BC5168-A518-4192-9D20-0B907B23C
interface IUser : IDispatch
{
    [propget, id(1), helpstring(hsIUser_Handle), helpcontext(15)] HRESULT Handle([out, retval] BSTR *pVal);
    [propput, id(1), helpstring(hsIUser_Handle), helpcontext(15)] HRESULT Handle([in] BSTR newValue);
    [propget, id(2), helpstring(hsIUser_FullName), helpcontext(15)] HRESULT FullName([out, retval] BSTR *pVal);
    [propget, id(3), helpstring(hsIUser_Birthday), helpcontext(15)] HRESULT Birthday([out, retval] DATE *pVal);
    [propget, id(4), helpstring(hsIUser_Sex), helpcontext(15)] HRESULT Sex([out, retval] TUserSex *pVal);
    [propget, id(5), helpstring(hsIUser_Country), helpcontext(15)] HRESULT Country([out, retval] BSTR *pVal);
    [propget, id(6), helpstring(hsIUser_Province), helpcontext(15)] HRESULT Province([out, retval] BSTR *pVal);
    [propget, id(7), helpstring(hsIUser_City), helpcontext(15)] HRESULT City([out, retval] BSTR *pVal);
    [propget, id(8), helpstring(hsIUser_PhoneHome), helpcontext(15)] HRESULT PhoneHome([out, retval] BSTR *pVal);
HRESULT PhoneOffice([out, retval] BSTR *pVal);
HRESULT PhoneMobile([out, retval] BSTR *pVal);
HRESULT Homepage([out, retval] BSTR *pVal);
HRESULT About([out, retval] BSTR *pVal);
HRESULT HasCallEquipment([out, retval] VARIANT_BOOL *pVal);
HRESULT BuddyStatus([out, retval] TBuddyStatus *pVal);
HRESULT IsAuthorized([out, retval] VARIANT_BOOL *pVal);
HRESULT IsBlocked([out, retval] VARIANT_BOOL *pVal);
HRESULT DisplayName([out, retval] BSTR *pVal);
HRESULT DisplayName([out, retval] BSTR *pVal);
HRESULT DisplayName([out, retval] BSTR *pVal);
HRESULT DisplayName([out, retval] BSTR *pVal);
HRESULT OnlineStatus([out, retval] TOnlineStatus *pVal);
HRESULT LastOnline([out, retval] DATE *pVal);
HRESULT CountryCode([out, retval] BSTR *pVal);
HRESULT ReceivedAuthRequest([out, retval] BSTR *pVal);
HRESULT SpeedDial([out, retval] BSTR *pVal);
HRESULT SpeedDial([in] BSTR newVal);
HRESULT CanLeaveVoiceMail([out, retval] VARIANT_BOOL *pVal);
HRESULT MoodText([out, retval] BSTR *pVal);
HRESULT Aliases([out, retval] BSTR *pVal);
HRESULT Timezone([out, retval] LONG *pVal);
HRESULT IsCallForwardActive([out, retval] VARIANT_BOOL *pVal);
HRESULT Language([out, retval] BSTR *pVal);
HRESULT LanguageCode([out, retval] BSTR *pVal);
e([out, retval] BSTR *pVal);
00653 [propget, id(30), helpstring(hsIUser_VideoCapable), helpcontext(15)] HRESULT IsVideoCapable([out, retval] VARIANT_BOOL *pVal);
00654 [propget, id(31), helpstring(hsIUser_SkypeOut), helpcontext(15)] HRESULT IsSkypeOutContact([out, retval] VARIANT_BOOL *pVal);
00655 [propget, id(32), helpstring(hsIUser_NumberOfAuthBuddies), helpcontext(15)] HRESULT NumberOfAuthBuddies([out, retval] LONG* pVal);
00656 [propput, id(17), helpstring(hsIUser_DisplayName), helpcontext(15)] HRESULT DisplayName([in] BSTR newVal);
00657 [propget, id(35), helpstring(hsIUser_RichMoodText), helpcontext(15)] HRESULT RichMoodText([out, retval] BSTR *pVal);
00658 [propget, id(36), helpstring(hsIUser_IsVmCapable), helpcontext(15)] HRESULT IsVoicemailCapable([out, retval] VARIANT_BOOL *pVal);
00659 
00660 [object,
00661 uuid(EC163200-44EB-483B-907F-A8C1CF56B8EE),
00662 dual,
00663 nonextensible,
00664 pointer_default(unique),
00665 oleautomation,
00666 helpstring(hsIUserColl),
00667 helpcontext(16)
00669 ]
00670 interface IUserCollection : IDispatch
00671 {
00672 [propget, id(1), helpstring(hsIUserColl_Count), helpcontext(16)] HRESULT Count([out, retval] LONG* pVal);
00673 [id(2), helpstring(hsIUserColl_Add), h
elpcontext(16) HRESULT Add([in] IUser* pUser);
00674 [id(3), helpstring(hsIUserColl_Remove), helpcontext(16)] HRESULT Remove([in] long Index);
00675 [id(4), helpstring(hsIUserColl_RemoveAll), helpcontext(16)] HRESULT RemoveAll(void);
00676 [propget, id(DISPID_VALUE), helpstring(hsIUserColl_Item), helpcontext(16)] HRESULT Item([in] LONG Index, [out, retval] IUser **pVal);
00677 [propget, id(DISPID_NEWENUM), helpstring(hsIUserColl_NewEnum), helpcontext(16)] HRESULT _NewEnum([out, retval] IUnknown** pVal);
00678 }
00679 [object,
00680 uuid(F1B2AD09-1961-4336-A6BF-93010B73DE72),
00681 dual,
00682 nonextensible,
00683 pointer_default(unique),
00685 oleautomation,
00686 helpstring(hsIParticipant),
00687 helpcontext(37)
00688 ]
00689 interface IParticipant : IDispatch
00690 {
00691 [propget, id(1), helpstring(hsIParticipant_Handle), helpcontext(37)] HRESULT Handle([out, retval] BSTR *pVal);
00692 [propget, id(2), helpstring(hsIParticipant_DispName), helpcontext(37)] HRESULT DisplayNa
00693 [propget, id(3), helpstring(hsIParticipant_CallType), helpcontext(37)] HRESULT CallType([out, retval] TCallType *pVal);
00694 [propget, id(4), helpstring(hsIParticipant_CallStatu
s([out, retval] TCallStatus *pVal);

object, uuid(F825FE12-9826-4BD2-BF7E-1ED95B8480FE),
dual, nonextensible, pointer_default(unique), oleautomation, helpstring(hsIParticipantColl), helpcontext(38)
]

interface IParticipantCollection : IDispatch
{
    [propget, id(1), helpstring(hsIParticipantColl_Count), helpcontext(38)] HRESULT Count([out, retval] LONG* pVal);
    [id(2), helpstring(hsIParticipantColl_Add), helpcontext(38)] HRESULT Add([in] IParticipant* pParticipant);
    [id(3), helpstring(hsIParticipantColl_Remove), helpcontext(38)] HRESULT Remove([in] long Index);
    [id(4), helpstring(hsIParticipantColl_RemoveAll), helpcontext(38)] HRESULT RemoveAll(void);
    [propget, id(DISPID_VALUE), helpstring(hsIParticipantColl_Item), helpcontext(38)] HRESULT Item([in] LONG Index, [out, retval] IParticipant ** pVal);
    [propget, id(DISPID_NEWENUM), helpstring(hsIParticipantColl_NewEnum), helpcontext(38)] HRESULT _NewEnum([out, retval] IUnknown** pVal);
}

[

interface ICall : IDispatch
{
    [propget, id(1), helpstring(hsICall_Id),helpcontext(5)] HRESULT Id([out, retval] LONG *pVal);
    [propget, id(2), helpstring(hsICall_Timestamp),helpcontext(5)] HRESULT Timestamp([out, retval] DATE *pVal);
    [propget, id(3), helpstring(hsICall_PartnerHandle),helpcontext(5)] HRESULT PartnerHandle([out, retval] BSTR *pVal);
    [propget, id(4), helpstring(hsICall_PartnerDisplayName),helpcontext(5)] HRESULT PartnerDisplayName([out, retval] BSTR *pVal);
    [propget, id(5), helpstring(hsICall_ConferenceId),helpcontext(5)] HRESULT ConferenceId([out, retval] LONG *pVal);
    [propget, id(6), helpstring(hsICall_Type),helpcontext(5)] HRESULT Type([out, retval] TCa llType *pVal);
    [propget, id(7), helpstring(hsICall_Status),helpcontext(5)] HRESULT Status([out, retval] TCa llStatus *pVal);
    [propput, id(7), helpstring(hsICall_Status),helpcontext(5)] HRESULT Status([in] TCa llStatus newVal);
    [propget, id(8), helpstring(hsICall_Fa
ilReason), helpcontext(5)) HRESULT FailureReason([out, retval] TCallFailureReason *pVal);
00736    [propget, id(9), helpstring(hsICall_Subject), helpcontext(5)] HRESULT Subject([out, retval] BSTR *pVal);
00737    [propget, id(10), helpstring(hsICall_PSTN_Number), helpcontext(5)] HRESULT PstnNumber([out, retval] BSTR *pVal);
00738    [propget, id(11), helpstring(hsICall_Duration), helpcontext(5)] HRESULT Duration([out, retval] LONG *pVal);
00739    [propget, id(12), helpstring(hsICall_PSTN_Status), helpcontext(5)] HRESULT PstnStatus([out, retval] BSTR *pVal);
00740    [propput, id(13), helpstring(hsICall_Seen), helpcontext(5)] HRESULT Seen([in] VARIANT_BOOL newVal);
00741    [id(14), helpstring(hsICall_Hold), helpcontext(5)] HRESULT Hold();
00742    [id(15), helpstring(hsICall_Resume), helpcontext(5)] HRESULT Resume();
00743    [id(16), helpstring(hsICall_Finish), helpcontext(5)] HRESULT Finish();
00744    [id(17), helpstring(hsICall_Answer), helpcontext(5)] HRESULT Answer();
00745    [propput, id(18), helpstring(hsICall_DTMF), helpcontext(5)] HRESULT DTMF([in] BSTR newVal);
00746    [propget, id(19), helpstring(hsICall_Participants), helpcontext(5)] HRESULT Participants([out, retval] IParticipantCollection **pVal);
00747    [id(20), helpstring(hsICall_Join), helpcontext(5)] HRESULT Join([in] LONG Id);
00748    [propget, id(21), helpstring(hsICall_VMDuration), helpcontext(5)] HRESULT VmDuration([out, retval] LONG *pVal);
00749    [propget, id(22), helpstring(hsICall_V
HRESULT VmAllowedDuration([out, retval] LONG *pVal);
HRESULT VideoStatus([out, retval] TCallVideoStatus *pVal);
HRESULT VideoSendStatus([out, retval] TCallVideoSendStatus *pVal);
HRESULT VideoReceiveStatus([out, retval] TCallVideoSendStatus *pVal);
HRESULT StartVideoSend();
HRESULT StopVideoSend();
HRESULT StartVideoReceive();
HRESULT StopVideoReceive();
HRESULT RedirectToVoicemail();
HRESULT Forward();
HRESULT Rate([out, retval] LONG *pVal);
HRESULT RateCurrency([out, retval] BSTR *pVal);
HRESULT RatePrecision([out, retval] LONG *pVal);
HRESULT InputDevice([in] TCallIoDeviceType DeviceType, [out, retval] BSTR* pVal);
HRESULT InputDevice([in] TCallIoDeviceType DeviceType, [out, retval] BSTR* pVal);
HRESULT InputDevice([in] TCallIoDeviceType DeviceType, [in] BSTR newVal);

HRESULT OutputDevice([in] TCallIoDeviceType DeviceType, [out, retval] BSTR* pVal);

HRESULT OutputDevice([in] TCallIoDeviceType DeviceType, [in] BSTR newVal);

HRESULT CaptureMicDevice([in] TCallIoDeviceType DeviceType, [out, retval] BSTR* pVal);

HRESULT CaptureMicDevice([in] TCallIoDeviceType DeviceType, [in] BSTR newVal);

HRESULT InputStatus([out, retval] VARIANT_BOOL* pVal);

HRESULT ForwardedBy([out, retval] BSTR* pVal);

HRESULT Seen([out, retval] VARIANT_BOOL *pVal);

HRESULT CanTransfer([in] BSTR Target, [out, retval] VARIANT_BOOL* pVal);

HRESULT TransferStatus([out, retval] TCallStatus* pVal);

HRESULT TransferActive([out, retval] VARIANT_BOOL* pVal);
HRESULT TransferredBy([out, retval] BSTR* pVal);
HRESULT TransferredTo([out, retval] BSTR* pVal);
HRESULT Transfer([in] BSTR Target);
HRESULT TargetIdentity([out, retval] BSTR *pVal);

interface ICallCollection : IDispatch
{
    HRESULT Count([out, retval] LONG* pVal);
    HRESULT Add([in] ICall* pCall);
    HRESULT Remove([in] long Index);
    HRESULT RemoveAll(void);
    HRESULT Item([i...]}
interface ICall

00796 [propget, id(DISPID_NEWENUM), helpstring(hsICallColl_NewEnum), helpcontext(6)] HRESULT _NewEnum([out, retval] IUnknown** pVal);

interface IChat;

00800 [object,
00801 uuid(4CFF5C70-3C95-4566-824A-FA164586D535),
00803 dual,
00804 nonextensible,
00805 pointer_default(unique),
00806 oleautomation,
00807 helpstring(hsIChatMessage),
00808 helpcontext(11)
00809 ]

interface IChatMessage : IDispatch
00811 {
00812 [propget, id(1), helpstring(hsIChatMsg_Id), helpcontext(11)] HRESULT Id([out, retval] LONG *pVal);
00813 [propget, id(2), helpstring(hsIChatMsg_Timestamp), helpcontext(11)] HRESULT Timestamp([out, retval] DATE *pVal);
00814 [propget, id(3), helpstring(hsIChatMsg_FromHandle), helpcontext(11)] HRESULT FromHandle([out, retval] BSTR *pVal);
00815 [propget, id(4), helpstring(hsIChatMsg_FromDispName), helpcontext(11)] HRESULT FromDisplayName([out, retval] BSTR *pVal);
00816 [propget, id(5), helpstring(hsIChatMsg_Type), helpcontext(11)] HRESULT Type([out, retval] TChatMessageType *pVal);
00817 [propget, id(6), helpstring(hsIChatMsg_Status), helpcontext(11)] HRESULT Status([out, retv
```c
TChatMessageStatus *pVal);
00818 [propget, id(7), helpstring(hsIChatMsgLeaveReason), helpcontext(11)] HRESULT LeaveReason([out, retval] TChatLeaveReason *pVal);
00819 [propget, id(8), helpstring(hsIChatMsgBody), helpcontext(11)] HRESULT Body([out, retval] BSTR *pVal);
00820 [propget, id(9), helpstring(hsIChatMsgChatName), helpcontext(11)] HRESULT ChatName([out, retval] BSTR *pVal);
00821 [propget, id(10), helpstring(hsIChatMsgUsers), helpcontext(11)] HRESULT Users([out, retval] IUserCollection **pVal);
00822 [propput, id(11), helpstring(hsIChatMsgSeen), helpcontext(11)] HRESULT Seen([in] VARIANT_BOOL newVal);
00823 [propget, id(12), helpstring(hsIChatMsgChat), helpcontext(11)] HRESULT Chat([out, retval] IChat** pVal);
00824 [propget, id(13), helpstring(hsIChatMsgSender), helpcontext(11)] HRESULT Sender([out, retval] IUser* *pVal);
00825 [propget, id(14), helpstring(hsIChatMsgEditedBy), helpcontext(11)] HRESULT EditedBy([out, retval] BSTR* pVal);
00826 [propget, id(15), helpstring(hsIChatMsgEditTime), helpcontext(11)] HRESULT EditedTimestamp([out, retval] DATE* pVal);
00827 [propput, id(8), helpstring(hsIChatMsgBody), helpcontext(11)] HRESULT Body([in] BSTR newValue);
00828 [propget, id(16), helpstring(hsIChatMsgRole), helpcontext(11)] HRESULT Role([out, retval] TChatMemberRole* pVal);
00829 [propget, id(17), helpstring(hsIChatMsgOptions), helpcontext(11)] HRESULT Options([out, retval] LONG* pVal);
```
HRESULT IsEditable([out, retval] VARIANT_BOOL* pVal);

interface IChatMessageCollection : IDispatch
{
    [propget, id(1), helpstring(hsIChatMsgColl_Count), helpcontext(12)] HRESULT Count([out, retval] LONG* pVal);
    [id(2), helpstring(hsIChatMsgColl_Add), helpcontext(12)] HRESULT Add([in] IChatMessage* pItem);
    [id(3), helpstring(hsIChatMsgColl_Remove), helpcontext(12)] HRESULT Remove([in] long Index);
    [id(4), helpstring(hsIChatMsgColl_RemoveAll), helpcontext(12)] HRESULT RemoveAll(void);
    [propget, id(DISPID_VALUE), helpstring(hsIChatMsgColl_Item), helpcontext(12)] HRESULT Item([in] LONG Index, [out, retval] IChatMessage** pVal);
    [propget, id(DISPID_NEWENUM), helpstring(hsIChatMsgColl_NewEnum), helpcontext(12)] HRESULT NewEnum([out, retval] IUnknown** pVal);
};
interface IChatMember : IDispatch
{
    [propget, id(1), helpstring(hsIChatMember_Id), helpcontext(85)] HRESULT Id([out, retval] BSTR* pVal);
    [propget, id(2), helpstring(hsIChatMember_Handle), helpcontext(85)] HRESULT Handle([out, retval] BSTR* pVal);
    [propget, id(3), helpstring(hsIChatMember_Role), helpcontext(85)] HRESULT Role([out, retval] TChatMemberRole* pVal);
    [propput, id(3), helpstring(hsIChatMember_Role), helpcontext(85)] HRESULT Role([in] TChatMemberRole newVal);
    [propget, id(4), helpstring(hsIChatMember_CanSetRole), helpcontext(85)] HRESULT CanSetRoleTo([in] TChatMemberRole Role, [out, retval] VARIANT_BOOL* pVal);
    [propget, id(5), helpstring(hsIChatMember_IsActive), helpcontext(85)] HRESULT IsActive([out, retval] VARIANT_BOOL* pVal);
    [propget, id(6), helpstring(hsIChatMember_Chat), helpcontext(85)] HRESULT Chat([out, retval] IChat** pVal);
};
[object,
 uuid(8FFC686B-2E5E-4763-8B03-FF4FD3A0C4DA),
 dual,
 pointer_default(unique),
 helpstring(hsIChatMemberCollection),
 helpcontext(86)
 ]

interface IChatMemberCollection : IDispatch
{
  [propget, id(1), helpstring(hsIChatMemberCollection_Count), helpcontext(86)] HRESULT Count([out, retval] LONG* pVal);
  [propget, id(DISPID_VALUE), helpstring(hsIChatMemberCollection_Item), helpcontext(86)] HRESULT Item([in] LONG Index, [out, retval] IChatMember ** pVal);
  [propget, id(DISPID_NEWENUM), helpstring(hsIChatMemberCollection_NewEnum), helpcontext(86)] HRESULT NewEnum([out, retval] IUnknown** pVal);
}

[object,
 uuid(8C24106C-3368-4580-93E5-5ED4ACCDE306),
 dual,
 nonextensible,
 pointer_default(unique),
 oleautomation,
 helpstring(hsIChat),
 helpcontext(31)
 ]
interface IChat : IDispatch
{
    [propget, id(1), helpstring(hsIChat_Name), helpcontext(31)] HRESULT Name([out, retval] BSTR *pVal);
    [propget, id(2), helpstring(hsIChat_Messages), helpcontext(31)] HRESULT Messages([out, retval] IChatMessageCollection **pVal);
    [propget, id(3), helpstring(hsIChat_Timestamp), helpcontext(31)] HRESULT Timestamp([out, retval] DATE *pVal);
    [propget, id(4), helpstring(hsIChat_Adder), helpcontext(31)] HRESULT Adder([out, retval] IUser **pVal);
    [propget, id(5), helpstring(hsIChat_Status), helpcontext(31)] HRESULT Status([out, retval] TChatStatus *pVal);
    [propget, id(6), helpstring(hsIChat_Posters), helpcontext(31)] HRESULT Posters([out, retval] IUserCollection **pVal);
    [propget, id(7), helpstring(hsIChat_Members), helpcontext(31)] HRESULT Members([out, retval] IUserCollection **pVal);
    [propget, id(8), helpstring(hsIChat_Topic), helpcontext(31)] HRESULT Topic([out, retval] BSTR *pVal);
    [propput, id(8), helpstring(hsIChat_Topic), helpcontext(31)] HRESULT Topic([in] BSTR new Val);
    [propget, id(9), helpstring(hsIChat_ActiveMembers), helpcontext(31)] HRESULT ActiveMembers([out, retval] IUserCollection **pVal);
    [propget, id(10), helpstring(hsIChat_FriendlyName), helpcontext(31)] HRESULT FriendlyName([out, retval] BSTR *pVal);
    [id(11), helpstring(hsIChat_OpenWindow), helpcontext(31)] HRESULT OpenWindow();
HRESULT SendMessage([in] BSTR MessageText, [out, retval] IChatMessage **pVal);
HRESULT Leave();
HRESULT AddMembers([in] IUserCollection* pMembers);
HRESULT RecentMessages([out, retval] IChatMessageCollection **pVal);
HRESULT Bookmarked([out, retval] VARIANT_BOOL *pVal);
HRESULT Bookmark();
HRESULT Unbookmark();
HRESULT TopicXML([out, retval] BSTR *pVal);
HRESULT TopicXML([in] BSTR newVal);
HRESULT MemberObjects([out, retval] IChatMemberCollection **pVal);
HRESULT Blob([out, retval] BSTR* pVal);
HRESULT Options([out, retval] LONG* pVal);
HRESULT Options([in] LONG newVal);
HRESULT PasswordHint([out, retval] BSTR* pVal);
HRESULT SetPassword([in] BSTR Password, [in, defaultvalue(""')] BSTR Hint);
HRESULT GuideLines([out, retval] BSTR* pVal);
HRESULT GuideLines([in] BSTR newVal);
HRESULT Description([out, retval] BSTR* pVal);
HRESULT Description([in] BSTR newVal);
HRESULT DialogPartner([out, retval] BSTR* pVal);
HRESULT ActivityTimes([out, retval] DATE* pVal);
HRESULT MyRole([out, retval] TChatMemberRole* pVal);
HRESULT Applicants([out, retval] IUserCollection** pVal);
HRESULT Join(void);
HRESULT Kick([in] BSTR Handle);
HRESULT KickBan([in] BSTR Handle);
HRESULT Disband(void);
HRESULT EnterPassword([in] BSTR Password);
HRESULT ClearRecentMessages(void);
HRESULT AcceptAdd(void);
HRESULT AlertString([in] BSTR newVal);
HRESULT Type([out, retval] TChatType* pVal);
HRESULT MyStatus([out, retval] TChatMyStatus* pVal);
};

interface IChatCollection : IDispatch {
    [propget, id(1), helpstring(hsIChatColl_Count),helpcontext(32)] HRESULT Count([out, retval] LONG* pVal);
    [id(2), helpstring(hsIChatColl_Add),he
lpcontext(32)] HRESULT Add([in] IChat* pItem);

00960   [id(3), helpstring(hsIChatColl_Remove)

,helpcontext(32)] HRESULT Remove([in] long Index);

00961   [id(4), helpstring(hsIChatColl_RemoveAll)

,helpcontext(32)] HRESULT RemoveAll(void);

00962   [propget, id(DISPID_VALUE), helpstring(hsIChatColl_Item),helpcontext(32)] HRESULT Item([in] LONG Index, [out, retval] IChat ** pVal);

00963   [propget, id(DISPID_NEWENUM), helpstring(hsIChatColl_NewEnum),helpcontext(32)] HRESULT _

NewEnum([out, retval] IUnknown** pVal);

00964   }

00965 

00966   [object,

00967   uuid(AEB48497-5090-479C-8BE0-BD51597156A1),

00968   dual,

00969   nonextensible,

00970   pointer_default(unique),

00971   oleautomation,

00972   helpstring(hsIConference),

00973   helpcontext(39)

00974   ]

00975   interface IConference : IDispatch

00976   {

00977   [propget, id(1), helpstring(hsIConference_Id), helpcontext(39)] HRESULT Id([out, retval] LONG *pVal);

00978   [propget, id(2), helpstring(hsIConference_Calls), helpcontext(39)] HRESULT Calls([out, retval] ICallCollection **pVal);

00979   [propget, id(3), helpstring(hsIConference_ActiveCalls), helpcontext(39)] HRESULT ActiveCalls([out, retval] ICallCollection **pVal);

00980   [id(4), helpstring(hsIConference_Hold)

, helpcontext(39)] HRESULT Hold();

00981   [id(5), helpstring(hsIConference_Resum}
HRESULT Resume();
HRESULT Finish();

interface IConferenceCollection : IDispatch {
    HRESULT Count([out, retval] LONG* pVal);
    HRESULT Add([in] IConference* pItem);
    HRESULT Remove([in] long Index);
    HRESULT RemoveAll(void);
    HRESULT Item([in] LONG Index, [out, retval] IConference* *pVal);
    HRESULT _NewEnum([out, retval] IUnknown** pVal);
}
interface IConversion : IDispatch
{
    [id(1), helpstring(hsIConversion_OnlineStatus), helpcontext(10)] HRESULT OnlineStatusToText([in] TOnlineStatus status, [out, retval] BSTR *pVal);
    [id(2), helpstring(hsIConversion_TxOnlineStatus), helpcontext(10)] HRESULT TextToOnlineStatus([in] BSTR Text, [out, retval] TOnlineStatus *pVal);
    [id(3), helpstring(hsIConversion_BuddyStatus), helpcontext(10)] HRESULT BuddyStatusToText([in] TBuddyStatus status, [out, retval] BSTR *pVal);
    [id(4), helpstring(hsIConversion_TxBuddyStatus), helpcontext(10)] HRESULT TextToBuddyStatus([in] BSTR Text, [out, retval] TBuddyStatus *pVal);
    [id(5), helpstring(hsIConversion_CallStatus), helpcontext(10)] HRESULT CallStatusToText([in] TCallStatus Status, [out, retval] BSTR *pVal);
    [id(6), helpstring(hsIConversion_TxCallStatus), helpcontext(10)] HRESULT TextToCallStatus([in] BSTR Text, [out, retval] TCallStatus *pVal)
HRESULT CallTypeToText([in] TCallType CallType, [out, retval] BSTR *pVal);
HRESULT TextToCallType([in] BSTR Text, [out, retval] TCallType *pVal);
HRESULT UserSexToText([in] TUserSex Sex, [out, retval] BSTR *pVal);
HRESULT TextToUserSex([in] BSTR Text, [out, retval] TUserSex *pVal);
HRESULT ConnectionStatusToText([in] TConnectionStatus Status, [out, retval] BSTR *pVal);
HRESULT TextToConnectionStatus([in] BSTR Text, [out, retval] TConnectionStatus *pVal);
HRESULT UserStatusToText([in] TUserStatus Status, [out, retval] BSTR *pVal);
HRESULT TextToUserStatus([in] BSTR Text, [out, retval] TUserStatus *pVal);
HRESULT CallFailureReasonToText([in] TCallFailureReason reason, [out, retval] BSTR *pVal);
HRESULT AttachmentStatusToText([in] TAttachmentStatus status, [out, retval] BSTR *pVal);
HRESULT ChatLeaveReasonToText([in] TChatLeaveReason reason, [out, retval] BSTR *pVal);

HRESULT ChatStatusToText([in] TChatStatus status, [out, retval] BSTR *pVal);

HRESULT VoicemailTypeToText([in] TVoicemailType type, [out, retval] BSTR *pVal);

HRESULT VoicemailStatusToText([in] TVoicemailStatus status, [out, retval] BSTR *pVal);

HRESULT TextToVoicemailStatus([in] BSTR Text, [out, retval] TVoicemailStatus *pVal);

HRESULT VoicemailFailureReasonToText([in] TVoicemailFailureReason code, [out, retval] BSTR *pVal);

HRESULT ChatMessageStatusToText([in] TChatMessageStatus Status, [out, retval] BSTR *pVal);

HRESULT TextToChatMessageStatus([in] BSTR Text, [out, retval] TChatMessageStatus *pVal);

HRESULT ChatMessageTypeToText([in] TChatMessageType Type, [out, retval] BSTR *pVal);

HRESULT ChatMessageStatusToText([in] TChatMessageStatus Type, [out, retval] BSTR *pVal);
ssType), helpcontext(10)) HRESULT TextToChatMessageType([in] BSTR Text, [out, retval] TChatMessageType *pVal);
01041 [id(27), helpstring(hsIConversion_TxAttachStat), helpcontext(10)) HRESULT TextToAttachmentStatus([in] BSTR Text, [out, retval] TAttachmentStatus *pVal);
01042 [id(28), helpstring(hsIConversion_GrpTypete), helpcontext(10)) HRESULT GroupTypeToText([in] TGroupType Type, [out, retval] BSTR *pVal);
01043 [id(29), helpstring(hsIConversion_TxGr pType), helpcontext(10)) HRESULT TextToGroupType([in] BSTR Text, [out, retval] TGroupType *pVal);
01044 [id(30), helpstring(hsIConversion_VideoStatus), helpcontext(10)) HRESULT CallVideoStatusToText([in] TCallVideoStatus Status, [out, retval] BSTR *pVal);
01045 [id(31), helpstring(hsIConversion_VideoSendStatus), helpcontext(10)) HRESULT CallVideoSendStatusToText([in] TCallVideoSendStatus Status, [out, retval] BSTR *pVal);
01046 [propput, id(32), helpstring(hsIConversion_Language), helpcontext(10)) HRESULT Language([in] BSTR newVal);
01047 [propput, id(32), helpstring(hsIConversion_Language), helpcontext(10)) HRESULT Language([in] BSTR newVal);
01048 [id(33), helpstring(hsIConversion_SmsS tatus), helpcontext(10)) HRESULT SmsMessageStatusToText([in] TSmsMessageStatus status, [out, retval] BSTR *pVal);
01049 [id(34), helpstring(hsIConversion_SmsT ype), helpcontext(10)) HRESULT SmsMessageTypeToText([in] TSmsMessageType type, [out, retval] BSTR *p Val);
01050 [id(35), helpstring(hsIConversion_SmsT argetStatus), helpcontext(10)) HRESULT SmsTargetSt...
statusToText([in] TSmsTargetStatus status, [out, retval] BSTR *pVal);
    
    [object,
    uuid(120B425E-6313-4924-B5A9-3E9F2E444A00),
    dual,
    nonextensible,
    pointer_default(unique),
    oleautomation,
    helpstring(hsIVoicemail),
    helpcontext(51)
    ]

    interface IVoicemail : IDispatch
    {
    [propget, id(1), helpstring(hsIVoicemail_Type), helpcontext(51)] HRESULT Type([out, retval] TVoicemailType *pVal);
    [propget, id(2), helpstring(hsIVoicemail_PartHandle), helpcontext(51)] HRESULT PartnerHandle([out, retval] BSTR *pVal);
    [propget, id(3), helpstring(hsIVoicemail_PartDispName), helpcontext(51)] HRESULT PartnerDisplayName([out, retval] BSTR *pVal);
    [propget, id(4), helpstring(hsIVoicemail_Status), helpcontext(51)] HRESULT Status([out, retval] TVoicemailStatus *pVal);
    [propget, id(5), helpstring(hsIVoicemail_FailureReason), helpcontext(51)] HRESULT FailureReason([out, retval] TVoicemailFailureReason *pVal);
    [propget, id(6), helpstring(hsIVoicemail_Timestamp), helpcontext(51)] HRESULT Timestamp([out, retval] DATE *pVal);
    [propget, id(7), helpstring(hsIVoicemail_Duration), helpcontext(51)] HRESULT Duration([o
allowedDuration([out, retval] LONG *pVal);
01072 [propget, id(9), helpstring(hsIVoicemail_Id), helpcontext(51)] HRESULT Id([out, retval] LONG *pVal);
01073 [id(10), helpstring(hsIVoicemail_Open), helpcontext(51)] HRESULT Open();
01074 [id(11), helpstring(hsIVoicemail_StartPlay), helpcontext(51)] HRESULT StartPlayback();
01075 [id(12), helpstring(hsIVoicemail_StopPlay), helpcontext(51)] HRESULT StopPlayback();
01076 [id(13), helpstring(hsIVoicemail_Upload), helpcontext(51)] HRESULT Upload();
01077 [id(14), helpstring(hsIVoicemail_Download), helpcontext(51)] HRESULT Download();
01078 [id(15), helpstring(hsIVoicemail_StartRec), helpcontext(51)] HRESULT StartRecording();
01079 [id(16), helpstring(hsIVoicemail_StopRec), helpcontext(51)] HRESULT StopRecording();
01080 [id(17), helpstring(hsIVoicemail_Delete), helpcontext(51)] HRESULT Delete();
01081 [id(18), helpstring(hsIVoicemail_PlayInCall), helpcontext(51)] HRESULT StartPlaybackInCall();
01082 [id(19), helpstring(hsIVoicemail_Unplayed), helpcontext(51)] HRESULT SetUnplayed();
01083 
01084 [object,
01085 uuid(21494848-BBD5-4192-BE32-57D1A19D8E94),
01086 dual,
01087 nonextensible,
01088 pointer_default(unique),
01089 oleautomation,
interface IVoicemailCollection : IDispatch {

    [propget, id(1), helpstring(hsIVoicemailColl_Count), helpcontext(52)] HRESULT Count([out, retval] LONG* pVal);

    [id(2), helpstring(hsIVoicemailColl_Add), helpcontext(52)] HRESULT Add([in] IVoicemail* pItem);

    [id(3), helpstring(hsIVoicemailColl_Remove), helpcontext(52)] HRESULT Remove([in] long Index);

    [id(4), helpstring(hsIVoicemailColl_RemoveAll), helpcontext(52)] HRESULT RemoveAll(void);

    [propget, id(DISPID_VALUE), helpstring(hsIVoicemailColl_Item), helpcontext(52)] HRESULT Item([in] LONG Index, [out, retval] IVoicemail ** pVal);

    [propget, id(DISPID_NEWENUM), helpstring(hsIVoicemailColl_NewEnum), helpcontext(52)] HRESULT _NewEnum([out, retval] IUnknown** pVal);

};
interface IApplicationStream : IDispatch
{
    [propget, id(1), helpstring(hsIAppStm_AppName), helpcontext(56)] HRESULT ApplicationName([out, retval] BSTR *pVal);
    [propget, id(2), helpstring(hsIAppStm_Handle), helpcontext(56)] HRESULT Handle([out, retval] BSTR *pVal);
    [id(3), helpstring(hsIAppStm_Read), helpcontext(56)] HRESULT Read([out, retval] BSTR *pText);
    [id(4), helpstring(hsIAppStm_Write), helpcontext(56)] HRESULT Write([in] BSTR Text);
    [id(5), helpstring(hsIAppStm_SendData), helpcontext(56)] HRESULT SendDatagram([in] BSTR Text);
    [propget, id(6), helpstring(hsIAppStm_DataLen), helpcontext(56)] HRESULT DataLength([out, retval] LONG *pVal);
    [id(7), helpstring(hsIAppStm_Disconnect), helpcontext(56)] HRESULT Disconnect();
    [propget, id(8), helpstring(hsIAppStm_PartnerHandle), helpcontext(56)] HRESULT PartnerHandle([out, retval] BSTR *pVal);
};

object,
    uuid(D8F1BCC4-2A73-47B9-83B3-D4D299CA5BC6),
    dual,
    nonextensible,
    pointer_default(unique),
    oleautomation,
    helpstring(hsIApplicationStreamColl),
    helpcontext(58)
];
interface IApplicationStreamCollection :
IDispatch
01136  
01137       [propget, id(1), helpstring(hsIAppStmColl_Count), helpcontext(58)] HRESULT Count([out, retval] LONG* pVal);
01138       [id(2), helpstring(hsIAppStmColl_Add), helpcontext(58)] HRESULT Add([in] IApplicationStream* pItem);
01139       [id(3), helpstring(hsIAppStmColl_Remove), helpcontext(58)] HRESULT Remove([in] long Index);
01140       [id(4), helpstring(hsIAppStmColl_RemoveAll), helpcontext(58)] HRESULT RemoveAll(void);
01141       [propget, id(DISPID_VALUE), helpstring(hsIAppStmColl_Item), helpcontext(58)] HRESULT Item ([in] LONG Index, [out, retval] IApplicationStream** pVal);
01142       [propget, id(DISPID_NEWENUM), helpstring(hsIAppStmColl_NewEnum), helpcontext(58)] HRESULT _NewEnum([out, retval] IUnknown** pVal);
01143     
01144     
01145      object,
01146      uuid(F9ADA565-2FAD-424D-878C-7AD12DF9E617),
01147      dual,
01148      nonextensible,
01149      pointer_default(unique),
01150      oleautomation,
01151      helpstring(hsIApplication),
01152      helpcontext(14)
01153    ]
01154  interface IApplication : IDispatch
01155  
01156      [propget, id(1), helpstring(hsIApp_Name), helpcontext(14)] HRESULT Name([out, retval] BSTR* pVal);
HRESULT Create();
HRESULT Delete();
HRESULT Streams([out, retval] IApplicationStreamCollection **pVal);
HRESULT ConnectableUsers([out, retval] IUserCollection **pVal);
HRESULT ConnectingUsers([out, retval] IUserCollection **pVal);
HRESULT Connect([in] BSTR Username, [in, defaultvalue(0)] VARIANT_BOOL WaitConnected);
HRESULT SendingStreams([out, retval] IApplicationStreamCollection **pVal);
HRESULT ReceivedStreams([out, retval] IApplicationStreamCollection **pVal);
HRESULT SendDatagram([in] BSTR Text, [in, defaultvalue(0)] IApplicationStreamCollection* pStreams);
interface IProfile : IDispatch
{
    [propget, id(1), helpstring(hsIProfile_Fullname), helpcontext(60)] HRESULT FullName([out, retval] BSTR *pVal);
    [propput, id(1), helpstring(hsIProfile_Fullname), helpcontext(60)] HRESULT FullName([in] BSTR newVal);
    [propget, id(2), helpstring(hsIProfile_Birthday), helpcontext(60)] HRESULT Birthday([out, retval] BSTR *pVal);
    [propput, id(2), helpstring(hsIProfile_Birthday), helpcontext(60)] HRESULT Birthday([in] BSTR newVal);
    [propget, id(3), helpstring(hsIProfile_Sex), helpcontext(60)] HRESULT Sex([out, retval] TUserSex *pVal);
    [propput, id(3), helpstring(hsIProfile_Sex), helpcontext(60)] HRESULT Sex([in] TUserSex newVal);
    [propget, id(4), helpstring(hsIProfile_Language), helpcontext(60)] HRESULT Languages([out, retval] BSTR *pVal);
    [propput, id(4), helpstring(hsIProfile_Language), helpcontext(60)] HRESULT Languages([in] BSTR newVal);
    [propget, id(5), helpstring(hsIProfile_Country), helpcontext(60)] HRESULT Country([out, retval] BSTR *pVal);
    [propput, id(5), helpstring(hsIProfile_Country), helpcontext(60)] HRESULT Country([in] BSTR newVal);
    [propget, id(6), helpstring(hsIProfile_Province), helpcontext(60)] HRESULT Province([out}
01190           [propput, id(6), helpstring(hsIProfile _Province), helpcontext(60)] HRESULT Province([in] BSTR newVal);
01191           [propget, id(7), helpstring(hsIProfile _City), helpcontext(60)] HRESULT City([out, retval] BSTR *pVal);
01192           [propput, id(7), helpstring(hsIProfile _City), helpcontext(60)] HRESULT City([in] BSTR newVal);
01193           [propget, id(8), helpstring(hsIProfile _PhoneHome), helpcontext(60)] HRESULT PhoneHome([out, retval] BSTR *pVal);
01194           [propput, id(8), helpstring(hsIProfile _PhoneHome), helpcontext(60)] HRESULT PhoneHome([in] BSTR newVal);
01195           [propget, id(9), helpstring(hsIProfile _PhoneOffice), helpcontext(60)] HRESULT PhoneOffice([out, retval] BSTR *pVal);
01196           [propput, id(9), helpstring(hsIProfile _PhoneOffice), helpcontext(60)] HRESULT PhoneOffice([in] BSTR newVal);
01197           [propget, id(10), helpstring(hsIProfile _PhoneMobile), helpcontext(60)] HRESULT PhoneMobile([out, retval] BSTR *pVal);
01198           [propput, id(10), helpstring(hsIProfile _PhoneMobile), helpcontext(60)] HRESULT PhoneMobile([in] BSTR newVal);
01199           [propget, id(11), helpstring(hsIProfile _Homepage), helpcontext(60)] HRESULT Homepage([out, retval] BSTR *pVal);
01200           [propput, id(11), helpstring(hsIProfile _Homepage), helpcontext(60)] HRESULT Homepage([in] BSTR newVal);
01201           [propget, id(12), helpstring(hsIProfile _About), helpcontext(60)] HRESULT About([out, retval] BSTR *pVal);
HRESULT About([in] BSTR newVal);

HRESULT MoodText([out, retval] BSTR *pVal);

HRESULT MoodText([in] BSTR newVal);

HRESULT Timezone([out, retval] LONG *pVal);

HRESULT Timezone([in] LONG newVal);

HRESULT CallNoAnswerTimeout([out, retval] short *pVal);

HRESULT CallNoAnswerTimeout([in] short newVal);

HRESULT CallApplyCF([out, retval] VARIANT_BOOL *pVal);

HRESULT CallApplyCF([in] VARIANT_BOOL newVal);

HRESULT CallSendToVM([out, retval] VARIANT_BOOL *pVal);

HRESULT CallSendToVM([in] VARIANT_BOOL newVal);

HRESULT CallForwardRules([out, retval] BSTR *pVal);

HRESULT CallForwardRules([in] BSTR newVal);
HRESULT CallForwardRules([in] BSTR newVal);

HRESULT Balance([out, retval] LONG *pVal);

HRESULT BalanceCurrency([out, retval] BSTR *pVal);

HRESULT BalanceToText([out, retval] BSTR *pVal);

HRESULT IPCountry([out, retval] BSTR *pVal);

HRESULT ValidateSmsNumbers([out, retval] BSTR* pVal);

HRESULT RichMoodText([out, retval] BSTR* pVal);

HRESULT RichMoodText([in] BSTR newVal);

interface IGroup : IDispatch

object uuid(84513274-9C49-4AAA-B8FC-316EB32CFE95),
dual, nonextensible,
pointer_default(unique), oleautomation,
helpstring(hsISmsGroup), helpcontext(9)

interface IGroup : IDispatch
HRESULT Id([out, retval] LONG *pVal);
HRESULT Type([out, retval] TGroupType *pVal);
HRESULT CustomGroupId([out, retval] BSTR *pVal);
HRESULT DisplayName([out, retval] BSTR *pVal);
HRESULT DisplayName([in] BSTR newVal);
HRESULT Users([out, retval] IUserCollection **pVal);
HRESULT OnlineUsers([out, retval] IUserCollection **pVal);
HRESULT IsVisible([out, retval] VARIANT_BOOL *pVal);
HRESULT IsExpanded([out, retval] VARIANT_BOOL *pVal);
HRESULT AddUser([in] BSTR Username);
HRESULT RemoveUser([in] BSTR Username);
HRESULT Share([in, defaultvalue(""')] BSTR MessageText);
HRESULT Accept,
HRESULT Accept();

HRESULT Decline();

interface IGroupCollection : IDispatch
{
    [propget, id(1), helpstring(hsIGroupCollection_Count), helpcontext(8)] HRESULT Count([out, retval] LONG* pVal);
    [id(2), helpstring(hsIGroupColl_Add), helpcontext(8)] HRESULT Add([in] IGroup* pGroup);
    [id(3), helpstring(hsIGroupColl_Remove), helpcontext(8)] HRESULT Remove([in] long Index);
    [id(4), helpstring(hsIGroupColl_RemoveAll), helpcontext(8)] HRESULT RemoveAll(void);
    [propget, id(DISPID_VALUE), helpstring(hsIGroupColl_Item), helpcontext(8)] HRESULT Item([in] LONG Index, [out, retval] IGroup ** pVal);
    [propget, id(DISPID_NEWENUM), helpstring(hsIGroupColl_NewEnum), helpcontext(8)] HRESULT _NewEnum([out, retval] IUnknown** pVal);
}

interface ISmsMessage;
interface ISmsChunk : IDispatch
{
    	[propget, id(1), helpstring(hsISmsChunk_Id), helpcontext(72)] HRESULT Id([out, retval] LONG *pVal);
    	[propget, id(2), helpstring(hsISmsChunk_CharsLeft), helpcontext(72)] HRESULT CharactersLeft([out, retval] LONG *pVal);
    	[propget, id(3), helpstring(hsISmsChunk_Text), helpcontext(72)] HRESULT Text([out, retval] BSTR *pVal);
    	[propget, id(4), helpstring(hsISmsChunk_Message), helpcontext(72)] HRESULT Message([out, retval] ISmsMessage** pVal);
};

interface ISmsChunkCollection : IDispatch
{
    	[propget, id(1), helpstring(hsISmsChunkCollection), helpcontext(73)] HRESULT Count([out, retval] long *pVal);
};
kColl_Count), helpcontext(73)] HRESULT Count([out, retval] LONG* pVal);
01300   [id(2), helpstring(hsISmsChunkColl_Add), helpcontext(73)] HRESULT Add([in] ISmsChunk* pItem);
01301   [id(3), helpstring(hsISmsChunkColl_Remove), helpcontext(73)] HRESULT Remove([in] long Index);
01302   [id(4), helpstring(hsISmsChunkColl_RemoveAll), helpcontext(73)] HRESULT RemoveAll(void);
01303   [propget, id(DISPID_VALUE), helpstring(hsISmsChunkColl_Item), helpcontext(73)] HRESULT Item([in] LONG Index, [out, retval] ISmsChunk ** pVal);
01304   [propget, id(DISPID_NEWENUM), helpstring(hsISmsChunkColl_NewEnum), helpcontext(73)] HRESULT _NewEnum([out, retval] IUnknown** pVal);
01305 }
01306   [object, uuid(A9B9C33A-49A6-41D5-B13F-2AA4F284F406),
01307 dual, nonextensible, pointer_default(unique), oleautomation, helpstring(hsISmsTarget), helpcontext(74)
01315 ]
01316 interface ISmsTarget : IDispatch
01317 {
01318   [propget, id(1), helpstring(hsISmsTarget_Status), helpcontext(74)] HRESULT Status([out, retval] TSmsTargetStatus *pVal);
01319   [propget, id(2), helpstring(hsISmsTarget_Number), helpcontext(74)] HRESULT Number([out, retval] BSTR *pVal);
HRESULT Message([out, retval] ISmsMessage** pVal);

object,

ISmsTargetCollection

interface ISmsTargetCollection : IDispatch

HRESULT Count([out, retval] LONG* pVal);

HRESULT Add([in] ISmsTarget* pItem);

HRESULT Remove([in] long Index);

HRESULT RemoveAll(void);

HRESULT Item([in] LONG Index, [out, retval] ISmsTarget** pVal);

HRESULT NewEnum([out, retval] IUnknown** pVal);
interface ISmsMessage : IDispatch
{
    [propget, id(1), helpstring(hsISmsMessage_Id), helpcontext(76)] HRESULT Id([out, retval] LONG *pVal);
    [propget, id(2), helpstring(hsISmsMessage_Type), helpcontext(76)] HRESULT Type([out, retval] TSmsMessageType *pVal);
    [propget, id(3), helpstring(hsISmsMessage_Status), helpcontext(76)] HRESULT Status([out, retval] TSmsMessageStatus *pVal);
    [propget, id(4), helpstring(hsISmsMessage_FailureReason), helpcontext(76)] HRESULT FailureReason([out, retval] TSmsFailureReason *pVal);
    [propget, id(5), helpstring(hsISmsMessage_FailedUnseen), helpcontext(76)] HRESULT IsFailedUnseen([out, retval] VARIANT_BOOL *pVal);
    [propput, id(6), helpstring(hsISmsMessage_Seen), helpcontext(76)] HRESULT Seen([in] VARIANT_BOOL newVal);
    [propget, id(7), helpstring(hsISmsMessage_Price), helpcontext(76)] HRESULT Price([out, retval] LONG *pVal);
    [propget, id(8), helpstring(hsISmsMessage_PricePrecision), helpcontext(76)] HRESULT PricePrecision([out, retval] LONG *pVal);
}
HRESULT PriceCurrency([out, retval] BSTR *pVal);
HRESULT ReplyToNumber([out, retval] BSTR *pVal);
HRESULT ReplyToNumber([in] BSTR newVal);
HRESULT Targets([out, retval] ISmsTargetCollection* *pVal);
HRESULT Body([out, retval] BSTR *pVal);
HRESULT Body([in] BSTR newVal);
HRESULT Chunks([out, retval] ISmsChunkCollection* *pVal);
HRESULT Timestamp([out, retval] DATE* pVal);
HRESULT Send();
HRESULT Delete();
HRESULT TargetNumbers([out, retval] BSTR* pVal);
HRESULT TargetNumbers([in] BSTR newVal);
interface ISmsMessageCollection : IDispatch {
    [propget, id(1), helpstring(hsISmsMessageColl_Count), helpcontext(77)] HRESULT Count([out, retval] LONG* pVal);
    [id(2), helpstring(hsISmsMessageColl_Add), helpcontext(77)] HRESULT Add([in] ISmsMessage* pItem);
    [id(3), helpstring(hsISmsMessageColl_Remove), helpcontext(77)] HRESULT Remove([in] long Index);
    [id(4), helpstring(hsISmsMessageColl_RemoveAll), helpcontext(77)] HRESULT RemoveAll(void);
    [propget, id(DISPID_VALUE), helpstring(hsISmsMessageColl_Item), helpcontext(77)] HRESULT Item([in] LONG Index, [out, retval] ISmsMessage** pVal);
    [propget, id(DISPID_NEWENUM), helpstring(hsISmsMessageColl_NewEnum), helpcontext(77)] HRESULT NewEnum([out, retval] IUnknown** pVal);
};
interface ISettings : IDispatch
{
   [propget, id(1), helpstring(hsISettings.AudioIn), helpcontext(2)] HRESULT AudioIn([out, retval] BSTR *pVal);
   [propput, id(1), helpstring(hsISettings.AudioIn), helpcontext(2)] HRESULT AudioIn([in] BSTR newVal);
   [propput, id(2), helpstring(hsISettings.AudioOut), helpcontext(2)] HRESULT AudioOut([in] BSTR newVal);
   [propput, id(3), helpstring(hsISettings.AutoAway), helpcontext(2)] HRESULT AutoAway([in] VARIANT_BOOL newVal);
   [propput, id(4), helpstring(hsISettings.Ringer), helpcontext(2)] HRESULT Ringer([in] BSTR newVal);
   [propput, id(5), helpstring(hsISettings.Avatar), helpcontext(2)] HRESULT Avatar([in, defaultvalue("1")], [in] BSTR Id, [in] BSTR newVal);
   [propput, id(6), helpstring(hsISettings)]]
HRESULT RingToneStatus([in, defaultvalue("1")]
BSTR Id,[out, retval]
VARIANT_BOOL *pVal);

HRESULT RingTone([in, defaultvalue("1")]
BSTR Id,[in] BSTR newVal);

HRESULT VideoIn([out, retval] BSTR *pVal);

HRESULT VideoIn([in] BSTR newVal);

HRESULT PCSpeaker([out, retval] VARIANT_BOOL *pVal);

HRESULT PCSpeaker([in] VARIANT_BOOL newVal);

HRESULT AGC([out, retval] VARIANT_BOOL *pVal);

HRESULT AGC([in] VARIANT_BOOL newVal);

HRESULT AEC([out, retval] VARIANT_BOOL *pVal);

HRESULT AEC([in] VARIANT_BOOL newVal);

HRESULT Language([out, retval] BSTR *pVal);

HRESULT Language([in] BSTR newVal);

HRESULT Language([in] BSTR newVal);
HRESULT RingToneStatus([in, defaultvalue("1")]) BSTR Id,[in] VARIANT_BOOL newVal);
01428   [id(13), helpstring(hsISettings_SaveAvatar), helpcontext(2)] HRESULT SaveAvatarToFile([in] BSTR Filename, [in, defaultvalue("1")]) BSTR AvatarId);
01429   [id(14), helpstring(hsISettings_LoadAvatar), helpcontext(2)] HRESULT LoadAvatarFromFile([in] BSTR Filename, [in, defaultvalue("1")]) BSTR AvatarId);
01430   [id(15), helpstring(hsISettings_IdleTimer),helpcontext(2)] HRESULT ResetIdleTimer(void);
01431 
01432   [object,
01433   uuid(C4BDF667-3FF7-4B44-A9F4-F3937E3E6D86),
01434   dual,
01435   nonextensible,
01436   pointer_default(unique),
01437   oleautomation,
01438   helpstring(hsIPluginMenuItem),
01439   helpcontext(87)
01440   ]
01441   interface IPluginMenuItem : IDispatch
01442   {
01443       [propget, id(1), helpstring(hsIPluginMenuItem_Id), helpcontext(87)] HRESULT Id([out, retval] BSTR* pVal);
01444       [propput, id(2), helpstring(hsIPluginMenuItem_Caption), helpcontext(87)] HRESULT Caption ([in] BSTR newVal);
01445       [propput, id(3), helpstring(hsIPluginMenuItem_Hint), helpcontext(87)] HRESULT Hint([in] BSTR newVal);
01446       [propput, id(4), helpstring(hsIPluginMenuItem_Hint), helpcontext(87)] HRESULT Hint([in] BSTR newVal);
01447       [propput, id(4), helpstring(hsIPluginMenuItem_Hint), helpcontext(87)] HRESULT Hint([in] BSTR newVal);
01448       [propput, id(4), helpstring(hsIPluginMenuItem_Hint), helpcontext(87)] HRESULT Hint([in] BSTR newVal);
01449       [propput, id(4), helpstring(hsIPluginMenuItem_Hint), helpcontext(87)] HRESULT Hint([in] BSTR newVal);
01450       [propput, id(4), helpstring(hsIPluginMenuItem_Hint), helpcontext(87)] HRESULT Hint([in] BSTR newVal);
01451       [propput, id(4), helpstring(hsIPluginMenuItem_Hint), helpcontext(87)] HRESULT Hint([in] BSTR newVal);
01452       [propput, id(4), helpstring(hsIPluginMenuItem_Hint), helpcontext(87)] HRESULT Hint([in] BSTR newVal);
01453       [propput, id(4), helpstring(hsIPluginMenuItem_Hint), helpcontext(87)] HRESULT Hint([in] BSTR newVal);
01454       [propput, id(4), helpstring(hsIPluginMenuItem_Hint), helpcontext(87)] HRESULT Hint([in] BSTR newVal);
01455       [propput, id(4), helpstring(hsIPluginMenuItem_Hint), helpcontext(87)] HRESULT Hint([in] BSTR newVal);
01456       [propput, id(4), helpstring(hsIPluginMenuItem_Hint), helpcontext(87)] HRESULT Hint([in] BSTR newVal);
01457   }
enuItem_Enabled), helpcontext(87)] HRESULT Enabled
([in] VARIANT_BOOL newVal);
01448 [id(5), helpstring(hsIPluginMenuItem_Delete), helpcontext(87)] HRESULT Delete(void);
01449 }
01450 [object,
01451 uuid(4DF5F83A-0ABA-417D-A6FC-62A68AE06EF7),
01452 dual,
01453 nonextensible,
01454 pointer_default(unique),
01455 oleautomation,
01456 helpstring(hsIPluginEvent),
01457 helpcontext(88)
01458 ]
01459 interface IPluginEvent : IDispatch
01460 {
01461 [propget, id(1), helpstring(hsIPluginEvent_Id),helpcontext(88)] HRESULT Id([out, retval] BSTR* pVal);
01462 [id(2), helpstring(hsIPluginEvent_Delete),helpcontext(88)] HRESULT Delete(void);
01463 }
01464 [object,
01465 uuid(4D36B368-B88C-45fa-B136-3EF77B2EA D39),
01466 dual,
01467 pointer_default(unique),
01468 helpstring(hsIFileTransfer),
01469 helpcontext(89)
01470 ]
01471 interface IFileTransfer : IDispatch
01472 {
01473 [propget, id(1), helpstring(hsIFileTransfer_Id), helpcontext(89)] HRESULT Id([out, retva
[propget, id(2), helpstring(hsIFileTransfer_Type), helpcontext(89)] HRESULT Type([out, retval] TFileTransferType* pVal);
01477 [propget, id(3), helpstring(hsIFileTransfer_Status), helpcontext(89)] HRESULT Status([out, retval] TFileTransferStatus* pVal);
01478 [propget, id(4), helpstring(hsIFileTransfer_FailReason), helpcontext(89)] HRESULT FailureReason([out, retval] TFileTransferFailureReason* pVal);
01479 [propget, id(5), helpstring(hsIFileTransfer_PartnerHandle), helpcontext(89)] HRESULT PartnerHandle([out, retval] BSTR* pVal);
01480 [propget, id(6), helpstring(hsIFileTransfer_PartnerDispName), helpcontext(89)] HRESULT PartnerDisplayName([out, retval] BSTR* pVal);
01481 [propget, id(7), helpstring(hsIFileTransfer_StartTime), helpcontext(89)] HRESULT StartTime([out, retval] DATE* pVal);
01482 [propget, id(8), helpstring(hsIFileTransfer_FinishTime), helpcontext(89)] HRESULT FinishTime([out, retval] DATE* pVal);
01483 [propget, id(9), helpstring(hsIFileTransfer_FilePath), helpcontext(89)] HRESULT FilePath([out, retval] BSTR* pVal);
01484 [propget, id(10), helpstring(hsIFileTransfer_FileName), helpcontext(89)] HRESULT FileName([out, retval] BSTR* pVal);
01485 [propget, id(11), helpstring(hsIFileTransfer_ByteSec), helpcontext(89)] HRESULT BytesPerSecond([out, retval] BSTR* pVal);
01486 [propget, id(12), helpstring(hsIFileTransfer_ByteTransferred), helpcontext(89)] HRESULT BytesTransferred([out, retval] BSTR* pVal);
interface IFileTransferCollection : IDispatch
{
    [propget, id(1), helpstring(hsIFileTransferColl_Count), helpcontext(90)] HRESULT Count([out, retval] LONG* pVal);
    [id(2), helpstring(hsIFileTransferColl_Add), helpcontext(90)] HRESULT Add([in] IFileTransfer* pItem);
    [id(3), helpstring(hsIFileTransferColl_Remove), helpcontext(90)] HRESULT Remove([in] long Index);
    [id(4), helpstring(hsIFileTransferColl_RemoveAll), helpcontext(90)] HRESULT RemoveAll(void);
    [propget, id(DISPID_VALUE), helpstring(hsIFileTransferColl_Item), helpcontext(90)] HRESULT Item([in] LONG Index, [out, retval] IFileTransfer** pVal);
    [propget, id(DISPID_NEWENUM), helpstring(hsIFileTransferColl_NewEnum), helpcontext(90)] HRESULT _NewEnum([out, retval] IUnknown** pVal);
};
interface IClient : IDispatch
{
    [id(1), helpstring(hsIClient_Start),helpcontext(36)] HRESULT Start([in, defaultvalue(0)] VARIANT_BOOL Minimized, [in, defaultvalue(0)] VARIANT_BOOL Nosplash);
    [id(2), helpstring(hsIClient_Minimize),helpcontext(36)] HRESULT Minimize();
    [id(3), helpstring(hsIClient_Shutdown),helpcontext(36)] HRESULT Shutdown();
    [propget, id(4), helpstring(hsIClient_Running),helpcontext(36)] HRESULT IsRunning([out, retval] VARIANT_BOOL *pVal);
    [id(5), helpstring(hsIClient_OpenProfile),helpcontext(36)] HRESULT OpenProfileDialog();
    [id(6), helpstring(hsIClient_OpenUserInfo),helpcontext(36)] HRESULT OpenUserInfoDialog([in] BSTR Username);
    [id(7), helpstring(hsIClient_OpenConference),helpcontext(36)] HRESULT OpenConferenceDialog();
    [id(8), helpstring(hsIClient_OpenSearch),helpcontext(36)] HRESULT OpenSearchDialog();
    [id(9), helpstring(hsIClient_OpenOptions),helpcontext(36)] HRESULT OpenOptionsDialog([in] BSTR Page);
    [id(10), helpstring(hsIClient_OpenCallHist),helpcontext(36)] HRESULT OpenCallHistoryTab();
    [id(11), helpstring(hsIClient_OpenContacts),helpcontext(36)] HRESULT OpenContactsTab();
    [id(12), helpstring(hsIClient_OpenDial}
01529 [id(13), helpstring(hsIClient_OpenSendCont), helpcontext(36)] HRESULT OpenSendContactsDialog([in, defaultvalue(""')] BSTR Username);
01530 [id(14), helpstring(hsIClient_OpenBlockedUser), helpcontext(36)] HRESULT OpenBlockedUsersDialog();
01531 [id(15), helpstring(hsIClient_OpenImportContact), helpcontext(36)] HRESULT OpenImportContactsWizard();
01532 [id(16), helpstring(hsIClient_OpenGettingStartedWizard), helpcontext(36)] HRESULT OpenGettingStartedWizard();
01533 [id(17), helpstring(hsIClient_OpenAuthorization), helpcontext(36)] HRESULT OpenAuthorizationDialog([in] BSTR Username);
01534 [id(18), helpstring(hsIClient_OpenDialog), helpcontext(36)] HRESULT OpenDialog([in] BSTR Name, [in, defaultvalue(""')] BSTR Param1, [in, defaultvalue(""')] BSTR Param2);
01535 [id(19), helpstring(hsIClient_OpenVideoTest), helpcontext(36)] HRESULT OpenVideoTestDialog();
01536 [id(20), helpstring(hsIClient_OpenAddContact), helpcontext(36)] HRESULT OpenAddContactDialog([in, defaultvalue(""')] BSTR Username);
01537 [id(21), helpstring(hsIClient_OpenMessage), helpcontext(36)] HRESULT OpenMessageDialog([in] BSTR Username, [in, defaultvalue(""')] BSTR Text);
01538 [id(22), helpstring(hsIClient_OpenFileTrans), helpcontext(36)] HRESULT OpenFileTransferDialog([in] BSTR User, [in] BSTR Folder);
01539 [id(23), helpstring(hsIClient_Focus), helpcontext(36)] HRESULT Focus();
01540 [id(24), helpstring(hsIClient_BtnPressed), helpcontext(36)] HRESULT ButtonPressed([in] BS
HRESULT ButtonReleased([in] BSTR Key);
HRESULT OpenSmsDialog([in] BSTR SmsId);
HRESULT CreateEvent([in] BSTR EventId, [in] BSTR Caption, [in] BSTR Hint, [out,retval] IPluginEvent** pEvent);
HRESULT CreateMenuItem([in] BSTR MenuItemId, [in] TPluginContext PluginContext, [in] BSTR CaptionText, [in,defaultvalue("") BSTR HintText, [in, defaultvalue("") BSTR IconPath, [in, defaultvalue(pluginContactTypeAll)] VARIANT_BOOL Enabled, [in, defaultvalue(pluginContactTypeAll)] TPluginContext ContactType, [in, defaultvalue(0)] VARIANT_BOOL MultipleContacts, [out,retval] IPluginMenuItem** pMenuItem);
HRESULT Wallpaper([out,retval] BSTR* pVal);
HRESULT Wallpaper([in] BSTR newVal);
HRESULT OpenLiveTab();
interface ISkype : IDispatch
{
    [propget, id(1), helpstring(hsISkype_Timeout), helpcontext(13)] HRESULT Timeout([out, retval] LONG *pVal);
    [propput, id(1), helpstring(hsISkype_Timeout), helpcontext(13)] HRESULT Timeout([in] LONG newVal);
    [propget, id(2), helpstring(hsISkype_Property), helpcontext(13)] HRESULT Property([in] BSTR ObjectType, [in] BSTR ObjectId, [in] BSTR PropName, [out, retval] BSTR *pVal);
    [propput, id(2), helpstring(hsISkype_Property), helpcontext(13)] HRESULT Property([in] BSTR ObjectType, [in] BSTR ObjectId, [in] BSTR PropName, [in] BSTR newVal);
    [propget, id(3), helpstring(hsISkype_Variable), helpcontext(13)] HRESULT Variable([in] BSTR Name, [out, retval] BSTR *pVal);
    [propput, id(3), helpstring(hsISkype_Variable), helpcontext(13)] HRESULT Variable([in] BSTR Name, [in] BSTR newVal);
    [propget, id(4), helpstring(hsISkype_CurrentUserHandle), helpcontext(13)] HRESULT CurrentUserHandle([out, retval] BSTR *pVal);
    [propput, id(5), helpstring(hsISkype_CurrentUserStatus), helpcontext(13)] HRESULT CurrentUserStatus([out, retval] TUserStatus *pVal);
    [propput, id(5), helpstring(hsISkype_CurrentUserStatus), helpcontext(13)] HRESULT CurrentUserStatus([in] TUserStatus newVal);
    [propget, id(6), helpstring(hsISkype_ConnectionStatus), helpcontext(13)] HRESULT ConnectionStatus([out, retval] TConnectionStatus *pVal);
HRESULT Mute([out, retval] VARIANT_BOOL *pVal);

HRESULT Mute([in] VARIANT_BOOL newVal);

HRESULT Version([out, retval] BSTR *pVal);

HRESULT Privilege([in] BSTR Name, [out, retval] VARIANT_BOOL *pVal);

HRESULT CurrentUser([out, retval] IUser **pVal);

HRESULT Convert([out, retval] IConversion **pVal);

HRESULT Friends([out, retval] IUserCollection **pVal);

HRESULT SearchForUsers([in] BSTR Target, [out, retval] IUserCollection **pVal);

HRESULT Calls([in, defaultvalue("")]) BSTR Target, [out, retval] ICallCollection **pVal);

HRESULT ActiveCalls([out, retval] ICallCollection **pVal);

HRESULT MissedCalls([out, retval] ICallCollection **pVal);

HRESULT Messages([in,
defaultvalue("")] BSTR Target, [out, retval] IChatMessageCollection **pVal);
01583    [propget, id(18), helpstring(hsISkype_MissedMessages), helpcontext(13)] HRESULT MissedMessages([out, retval] IChatMessageCollection **pVal);
01584    [propget, id(19), helpstring(hsISkype_AttachmentStatus), helpcontext(13)] HRESULT AttachmentStatus([out, retval] IAttachmentStatus *pVal);
01585    [propget, id(20), helpstring(hsISkype_Protocol), helpcontext(13)] HRESULT Protocol([out, retval] LONG *pVal);
01586    [propput, id(20), helpstring(hsISkype_Protocol), helpcontext(13)] HRESULT Protocol([in] LONG newVal);
01587    [id(21), helpstring(hsISkype_Attach), helpcontext(13)] HRESULT Attach([in, defaultvalue(5)] LONG Protocol, [in, defaultvalue(-1)] VARIANT_BOOL Wait);
01588    [id(22), helpstring(hsISkype_PlaceCall), helpcontext(13)] HRESULT PlaceCall([in] BSTR Target, [in, defaultvalue("")) BSTR Target2, [in, defaultvalue("")) BSTR Target3, [in, defaultvalue("")) BSTR Target4, [out, retval] ICall **pCall);
01589    [id(23), helpstring(hsISkype_SendMessage), helpcontext(13)] HRESULT SendMessage([in] BSTR Username, [in] BSTR Text, [out, retval] IChatMessage **pMessage);
01590    [propget, id(24), helpstring(hsISkype_User), helpcontext(13)] HRESULT User([in, defaultvalue("")) BSTR Username, [out, retval] IUser **pVal);
01591    [propget, id(25), helpstring(hsISkype_Message), helpcontext(13)] HRESULT Message([in, defaultvalue(0)] LONG Id, [out, retval] IChatMessage **pVal);
01592    [propget, id(26), helpstring(hsISkype_
HRESULT Call([in, defaultvalue(0)] LONG Id, [out, retval] ICall **pVal);
01593 [id(27), helprstring(hsISkype_SendCommand), helpcontext(13)] HRESULT SendCommand([in] ICommand* pCommand);
01594 [propget, id(28), helprstring(hsISkype_Chats), helpcontext(13)] HRESULT Chats([out, retval] IChatCollection **pVal);
01595 [propget, id(29), helprstring(hsISkype_Chat), helpcontext(13)] HRESULT Chat([in, defaultvalue("")) BSTR Name, [out, retval] IChat **pVal);
01596 [id(30), helprstring(hsISkype_ChangeUserStatus), helpcontext(13)] HRESULT ChangeUserStatus([in] TUserStatus newVal);
01597 [propget, id(31), helprstring(hsISkype_Conference), helpcontext(13)] HRESULT Conference([in, defaultvalue(0)] LONG Id,[out, retval] IConference **pVal);
01598 [propget, id(32), helprstring(hsISkype_Conferences), helpcontext(13)] HRESULT Conferences ([out, retval] IConferenceCollection **pVal);
01599 [propget, id(33), helprstring(hsISkype_Profile), helpcontext(13)] HRESULT Profile([in] BSTR Property, [out, retval] BSTR *pVal);
01600 [propput, id(33), helprstring(hsISkype_Profile), helpcontext(13)] HRESULT Profile([in] BSTR Property, [in] BSTR newVal);
01601 [propget, id(34), helprstring(hsISkype_ActiveChats), helpcontext(13)] HRESULT ActiveChats ([out, retval] IChatCollection **pVal);
01602 [propget, id(35), helprstring(hsISkype_MissedChats), helpcontext(13)] HRESULT MissedChats ([out, retval] IChatCollection **pVal);
01603 [propget, id(36), helprstring(hsISkype_RecentChats), helpcontext(13)] HRESULT RecentChats ([out, retval] IChatCollection **pVal);
HRESULT BookmarkedChats([out, retval] IChatCollection **pVal);
HRESULT CreateChatWith([in] BSTR Username, [out, retval] IChat **pVal);
HRESULT CreateChatMultiple([in] IUserCollection *pMembers, [out, retval] IChat **pChat);
HRESULT Voicemails([out, retval] IVoicemailCollection **pVal);
HRESULT SendVoicemail([in] BSTR Username, [out, retval] IVoicemail **pVoicemail);
HRESULT UsersWaitingAuthorization([out, retval] IUserCollection **pVal);
HRESULT ClearChatHistory();
HRESULT ClearVoicemailHistory();
HRESULT ClearCallHistory([in, defaultvalue("ALL")]) BSTR Username, [in, defaultvalue(chsAllCalls)] TCallHistory type);
HRESULT CommandId([out, retval] VARIANT_BOOL *pVal);
HRESULT CommandId([in] VARIANT_BOOL newVal);
HRESULT Application([in] BSTR Name, [out, retval] IApplication **pVal);

HRESULT Greeting([in, defaultvalue("")],[out, retval] IVoiceemail **pVal);

HRESULT Cache([out, retval] VARIANT_BOOL *pVal);

HRESULT Cache([in] VARIANT_BOOL newVal);

HRESULT ResetCache();

HRESULT CurrentUserProfile([out, retval] IProfile **pVal);

HRESULT Groups([out, retval] IGroupCollection* *pVal);

HRESULT CustomGroups([out, retval] IGroupCollection* *pVal);

HRESULT HardwiredGroups([out, retval] IGroupCollection* *pVal);

HRESULT CreateGroup([in] BSTR GroupName, [out, retval] IGroup** pGroup);

HRESULT DeleteGroup([in] LONG GroupId);

HRESULT Settings([out, retval] ISettings* *pVal);
HRESULT Client([out, retval] IClient** pVal);

HRESULT FriendlyName([in] BSTR newVal);

HRESULT Command([in] LONG Id, [in] BSTR Command, [in, defaultvalue("" )] BSTR Reply, [in, defaultvalue(0)] VARIANT_BOOL Block, [in, defaultvalue(30000)] LONG Timeout, [out, retval] ICommand** pVal);

HRESULT Voicemail([in] LONG Id, [out, retval] IVoicemail** pVal);

HRESULT MissedVoicemails([out, retval] IVoicemailCollection** pVal);

HRESULT EnableApiSecurityContext([in] TApiSecurityContext Context);

HRESULT ApiSecurityContextEnabled([in] TApiSecurityContext Context, [out, retval] VARIANT_BOOL* pVal);

HRESULT CreateSms([in] TSmsMessageType MessageType, [in] BSTR TargetNumbers, [out, retval] ISmsMessage** pMessage);

HRESULT Smss([out, retval] ISmsMessageCollection** pVal);

HRESULT MissedSmss([out, retval] ISmsMessageCollection** pVal);

HRESULT SendSms,
HRESULT SendSms([in] BSTR Target Numbers, [in] BSTR MessageText, [in, defaultvalue(""')] BSTR ReplyToNumber, [out, retval] ISmsMessage ** pMessage);

HRESULT AsyncSearchUsers([in] BSTR Target, [out, retval] LONG* pCookie);

HRESULT ApiWrapperVersion([out, retval] BSTR* pVal);

HRESULT SilentMode([in] VARIANT_BOOL newVal);

HRESULT FileTransfers([out, retval] IFileTransferCollection **pVal);

HRESULT ActiveFileTransfers([out, retval] IFileTransferCollection **pVal);

HRESULT FocusedContacts([out, retval] IUserCollection **pVal);

HRESULT FindChatUsingBlob([in] BSTR Blob, [out, retval] IChat** pChat);

HRESULT CreateChatUsingBlob([in] BSTR Blob, [out, retval] IChat** pChat);

HRESULT PredictiveDialerCountry([out] BSTR *pVal);
ifndef DOXYGEN_SHOULD_SKIP_THIS
[
object,
uuid(6D0B4547-771D-4C4F-B3E8-98A33FD24D2B),
dual,
nonextensible,
pointer_default(unique),
oleautomation,
helpstring(hsISkypeSms),
helpcontext(78),
hidden
] interface ISkypeSms : IDispatch
{
[propget, id(1), helpstring(hsISmsMessageType), helpcontext(78)] HRESULT SmsMessageType([in] BSTR SmsId, [out, retval] TSmsMessageType *pVal);
[propget, id(2), helpstring(hsISmsMessageStatus), helpcontext(78)] HRESULT SmsMessageStatus([in] BSTR SmsId, [out, retval] TSmsMessageStatus *pVal);
[propget, id(3), helpstring(hsISmsMessageFailureReason), helpcontext(78)] HRESULT SmsMessageFailureReason([in] BSTR SmsId, [out, retval] TSmsMessageFailureReason *pVal);
[propget, id(4), helpstring(hsISmsMessageFailedUnseen), helpcontext(78)] HRESULT SmsIsFailedUnseen([in] BSTR SmsId, [out, retval] VARIANT_BOOL *pVal);
[propput, id(5), helpstring(hsISmsMessageSeen), helpcontext(78)] HRESULT SmsSeen([in] BSTR SmsId);
[propget, id(6), helpstring(hsISmsMessagePrice), helpcontext(78)] HRESULT SmsPrice([in] BSTR SmsId, [out, retval] LONG *pVal);
HRESULT SmsPriceCurrency([in] BSTR SmsId, [out, retval] BSTR *pVal);
HRESULT SmsPricePrecision([in] BSTR SmsId, [out, retval] LONG *pVal);
HRESULT SmsReplyToNumber([in] BSTR SmsId, [out, retval] BSTR *pVal);
HRESULT SmsTargets([in] BSTR SmsId, [out, retval] ISmsTargetCollection** pVal);
HRESULT SmsBody([in] BSTR SmsId, [out, retval] BSTR *pVal);
HRESULT SmsChunks([in] BSTR SmsId, [out, retval] ISmsChunkCollection** pVal);
HRESULT SendSmsMessage([in] BSTR SmsId);
HRESULT DeleteSmsMessage([in] BSTR SmsId);
HRESULT SmsTargetStatus([in] BSTR SmsId, [in] BSTR Number, [out, retval] BSTR *pVal);
1] TSmsTargetStatus* pVal);
01680    [propget, id(16), helpstring(hsISmsChunk_Text), helpcontext(78)] HRESULT SmsChunkText([in] BSTR SmsId, [in] BSTR ChunkId, [out, retval] BSTR* pVal);
01681    [propget, id(17), helpstring(hsISmsChunk_CharsLeft), helpcontext(78)] HRESULT SmsChunkCharactersLeft([in] BSTR SmsId, [in] BSTR ChunkId, [out, retval] LONG* pVal);
01682    [propget, id(18), helpstring(hsISmsMessage_Timestamp), helpcontext(78)] HRESULT SmsTimestamp([in] BSTR SmsId, [out, retval] DATE* pVal);
01683    [propget, id(19), helpstring(hsISmsMessage_TargetNumbers), helpcontext(78)] HRESULT SmsTargetNumbers([in] BSTR SmsId, [out, retval] BSTR* pVal);
01684    [propput, id(19), helpstring(hsISmsMessage_TargetNumbers), helpcontext(78)] HRESULT SmsTargetNumbers([in] BSTR SmsId, [in] BSTR newVal);
01685    };
01686
01687    object,
01688    uuid(54590FC3-04D5-4A2D-B4A5-BEAC02657F2),
01689    dual,
01690    nonextensible,
01691    pointer_default(unique),
01692    oleautomation,
01693    helpstring(hsISkypeCall),
01694    helpcontext(45),
01695    hidden
01696    ]
01697    interface ISkypeCall : IDispatch
01698    {
01699    [propget, id(1), helpstring(hsICall_Timestamp), helpcontext(45), ] HRESULT CallTimestamp ([in] BSTR CallId, [out, retval] DATE *pVal);
HRESULT CallPartnerHandle([in] BSTR CallId, [out, retval] BSTR *pVal);

HRESULT CallPartnerDispName([in] BSTR CallId, [out, retval] BSTR *pVal);

HRESULT CallConferenceId([in] BSTR CallId, [out, retval] BSTR *pVal);

HRESULT CallType([in] BSTR CallId, [out, retval] TCallType *pVal);

HRESULT CallStatus([in] BSTR CallId, [in] TCallStatus newVal);

HRESULT CallFailureReason([in] BSTR CallId, [out, retval] TCallFailureReason *pVal);

HRESULT CallSubject([in] BSTR CallId, [out, retval] BSTR *pVal);

HRESULT CallPstnNumber([in] BSTR CallId, [out, retval] BSTR *pVal);

HRESULT CallDuration([in] BSTR CallId, [out, retval] LONG *pVal);

HRESULT CallPstnStatus([in] BSTR CallId, [out, retval] BSTR *pVal);

HRESULT CallSeenStatus([in] BSTR CallId, [out, retval] BSTR *pVal);
BSTR CallId, [in] VARIANT_BOOL newVal);
01712   [id(13), helpstring(hsICall_Hold), helpcontext(45)] HRESULT CallHold([in] BSTR CallId);
01713   [id(14), helpstring(hsICall_Resume), helpcontext(45)] HRESULT CallResume([in] BSTR CallId);
01714   [id(15), helpstring(hsICall_Finish), helpcontext(45)] HRESULT CallFinish([in] BSTR CallId);
01715   [id(16), helpstring(hsICall_Answer), helpcontext(45)] HRESULT CallAnswer([in] BSTR CallId);
01716   [propput, id(17), helpstring(hsICall_DTMF), helpcontext(45)] HRESULT CallDTMF([in] BSTR CallId, [in] BSTR newVal);
01717   [propget, id(18), helpstring(hsICall_Participants), helpcontext(45)] HRESULT CallParticipants([in] BSTR CallId, [out, retval] IParticipantCollection **pVal);
01718   [id(19), helpstring(hsICall_Join), helpcontext(45)] HRESULT CallJoin([in] BSTR CallId1, [in] BSTR CallId2);
01719   [propget, id(20), helpstring(hsICall_VmDuration), helpcontext(45)] HRESULT CallVmDuration([in] BSTR CallId, [out, retval] LONG *pVal);
01720   [propget, id(21), helpstring(hsICall_VmAllowDuration), helpcontext(45)] HRESULT CallVmAllowedDuration([in] BSTR CallId, [out, retval] LONG *pVal);
01721   [propget, id(22), helpstring(hsICall_VStatus), helpcontext(45)] HRESULT CallVideoStatus([in] BSTR CallId, [out, retval] TCallVideoStatus *pVal);
01722   [propget, id(23), helpstring(hsICall_VSendStatus), helpcontext(45)] HRESULT CallVideoSendStatus([in] BSTR CallId, [out, retval] TCallVideoSendStatus *pVal);
HRESULT CallVideoReceiveStatus([in] BSTR CallId, [out, retval] TCallVideoSendStatus *pVal);
HRESULT StartCallVideoSend([in] BSTR CallId);
HRESULT StopCallVideoSend([in] BSTR CallId);
HRESULT StartCallVideoReceive([in] BSTR CallId);
HRESULT StopCallVideoReceive([in] BSTR CallId);
HRESULT RedirectCallToVoicemail([in] BSTR CallId);
HRESULT ForwardCall([in] BSTR CallId);
HRESULT CallRate([in] BSTR CallId, [out, retval] LONG *pVal);
HRESULT CallRateCurrent([in] BSTR CallId, [out, retval] BSTR *pVal);
HRESULT CallRatePrecision([in] BSTR CallId, [out, retval] LONG *pVal);
HRESULT CallInputDevice([in] BSTR CallId, [in] TCallIoDeviceType DeviceType, [out, retval] BSTR* pVal);
HRESULT CallInputDevice([in] BSTR CallId, [in] TCallIoDeviceType DeviceType, [in] BSTR newVal);
01735  [propget, id(35), helpstring(hsICall_OutputDevice), helpcontext(45)] HRESULT CallOutputDevice([in] BSTR CallId, [in] TCallIoDeviceType DeviceType, [out, retval] BSTR* pVal);
01736  [propput, id(35), helpstring(hsICall_OutputDevice), helpcontext(45)] HRESULT CallOutputDevice([in] BSTR CallId, [in] TCallIoDeviceType DeviceType, [in] BSTR newVal);
01737  [propget, id(36), helpstring(hsICall_MicDevice), helpcontext(45)] HRESULT CallCaptureMicDevice([in] BSTR CallId, [in] TCallIoDeviceType DeviceType, [out, retval] BSTR* pVal);
01738  [propput, id(36), helpstring(hsICall_MicDevice), helpcontext(45)] HRESULT CallCaptureMicDevice([in] BSTR CallId, [in] TCallIoDeviceType DeviceType, [in] BSTR newVal);
01739  [propget, id(37), helpstring(hsICall_InputStatus), helpcontext(45)] HRESULT CallInputStatus([in] BSTR CallId, [out, retval] VARIANT_BOOL* pVal);
01740  [propget, id(38), helpstring(hsICall_ForwardedBy), helpcontext(45)] HRESULT CallForwardedBy([in] BSTR CallId, [out, retval] BSTR* pVal);
01741  [propget, id(12), helpstring(hsICall_Seen), helpcontext(45)] HRESULT CallSeenStatus([in] BSTR CallId, [out, retval] VARIANT_BOOL *pVal);
01742  [propget, id(39), helpstring(hsICall_CanTransfer), helpcontext(45)] HRESULT CallCanTransfer([in] BSTR CallId, [in] BSTR Target, [out, retval] VARIANT_BOOL *pVal);
01743  [propget, id(40), helpstring(hsICall_TransferStatus), helpcontext(45)] HRESULT CallTransferStatus([in] BSTR CallId, [out, retval] TCallStatus *pVal);
HRESULT CallTransferActive([in] BSTR CallId, [out, retval] VARIANT_BOOL *pVal);

HRESULT CallTransferredBy([in] BSTR CallId, [out, retval] BSTR *pVal);

HRESULT CallTransferredTo([in] BSTR CallId, [out, retval] BSTR *pVal);

HRESULT TransferCall([in] BSTR CallId, [in] BSTR Target);

HRESULT CallTargetIdentity([in] BSTR CallId, [out, retval] BSTR *pVal);

interface ISkypeUser : IDispatch
{
    [propget, id(1), helpstring(hsIUser_FullName), helpcontext(46)] HRESULT UserFullName([in] BSTR Username, [out, retval] BSTR *pVal);
HRESULT UserBirthday([in] BSTR Username, [out, retval] DATE *pVal);
HRESULT UserSex([in] BSTR Username, [out, retval] TUserSex *pVal);
HRESULT UserCountry([in] BSTR Username, [out, retval] BSTR *pVal);
HRESULT UserProvince([in] BSTR Username, [out, retval] BSTR *pVal);
HRESULT UserCity([in] BSTR Username, [out, retval] BSTR *pVal);
HRESULT UserPhoneHome([in] BSTR Username, [out, retval] BSTR *pVal);
HRESULT UserPhoneOffice([in] BSTR Username, [out, retval] BSTR *pVal);
HRESULT UserPhoneMobile([in] BSTR Username, [out, retval] BSTR *pVal);
HRESULT UserHomepage([in] BSTR Username, [out, retval] BSTR *pVal);
HRESULT UserAbout([in] BSTR Username, [out, retval] BSTR *pVal);
VARIANT_BOOL UserHasCallEquipment([in] BSTR Username, [out, retval] VARIANT_BOOL *pVal);
TBuddyStatus UserBuddyStatus([in] BSTR Username, [out, retval] TBuddyStatus *pVal);
HRESULT UserBuddyStatus([in] BSTR Username, [in] TBuddyStatus newVal);
HRESULT UserIsAuthorized([in] BSTR Username, [in] VARIANT_BOOL newVal);
HRESULT UserIsBlocked([in] BSTR Username, [in] VARIANT_BOOL *pVal);
HRESULT UserIsAuthorized([in] BSTR Username, [in] BSTR *pVal);
HRESULT UserDisplayName([in] BSTR Username, [out, retval] BSTR *pVal);
HRESULT UserOnlineStatus([in] BSTR Username, [out, retval] TOnlineStatus *pVal);
HRESULT UserLastOnline([in] BSTR Username, [out, retval] DATE *pVal);
HRESULT UserCountryCode([in] BSTR Username, [out, retval] BSTR *pVal);
HRESULT UserReceivedAuthRequest([in] BSTR Username, [out, retval] BSTR *pVal);
HRESULT UserSpeedDial([in] BSTR Username, [out, retval] BSTR *pVal);
[in] BSTR Username, [out, retval] BSTR *pVal);
01787 [propput, id(21), helpstring(hsIUser_SpeedDial), helpcontext(46)] HRESULT UserSpeedDial(
[in] BSTR Username, [in] BSTR newVal);
01788 [propget, id(22), helpstring(hsIUser_CanLeaveVm), helpcontext(46)] HRESULT UserCanLeaveVoiceMail([in] BSTR Username, [out, retval] VARIANT_BOOL *pVal);
01789 [propget, id(23), helpstring(hsIUser_MoodText), helpcontext(46)] HRESULT UserMoodText([in] BSTR Username, [out, retval] BSTR *pVal);
01790 [propget, id(24), helpstring(hsIUser_Aliases), helpcontext(46)] HRESULT UserAliases([in] BSTR Username, [out, retval] BSTR *pVal);
01791 [propget, id(25), helpstring(hsIUser_Timezone), helpcontext(46)] HRESULT UserTimezone([in] BSTR Username, [out, retval] LONG *pVal);
01792 [propget, id(26), helpstring(hsIUser_IsCFActive), helpcontext(46)] HRESULT IsCallForwardActive([in] BSTR Username, [out, retval] VARIANT_BOOL *pVal);
01793 [propget, id(27), helpstring(hsIUser_Language), helpcontext(46)] HRESULT UserLanguage([in] BSTR Username, [out, retval] BSTR *pVal);
01794 [propget, id(28), helpstring(hsIUser_LanguageCode), helpcontext(46)] HRESULT UserLanguageCode([in] BSTR Username, [out, retval] BSTR *pVal);
01795 [propget, id(29), helpstring(hsIUser_VideoCapable), helpcontext(46)] HRESULT UserIsVideoCapable([in] BSTR Username, [out, retval] VARIANT_BOOL *pVal);
01796 [propget, id(30), helpstring(hsIUser_NumberOfAuthBuddies), helpcontext(46)] HRESULT UserNumberOfAuthBuddies([in] BSTR Username, [out, retval] LONG* pVal);
01797 [propput, id(16), helpstring(hsIUser_
HRESULT UserDisplayName([in] BSTR Username, [in] BSTR newVal);

HRESULT UserRichMoodText([in] BSTR Username, [out, retval] BSTR *pVal);

HRESULT UserIsVoiceMailCapable([in] BSTR Username, [out, retval] VARIANT_BOOL *pVal);

interface ISkypeChatMessage : IDispatch
{
    [propget, id(1), helpstring(hsIChatMsg_Timestamp), helpcontext(47)] HRESULT ChatMessageTimestamp([in] BSTR MessageId, [out, retval] DATE *pVal);
    [propget, id(2), helpstring(hsIChatMsg_FromHandle), helpcontext(47)] HRESULT ChatMessageFromHandle([in] BSTR MessageId, [out, retval] BSTR *pVal);
    [propget, id(3), helpstring(hsIChatMsg_FromDispName), helpcontext(47)] HRESULT ChatMessageFromDisplayName([in] BSTR MessageId, [out, retval] BSTR *pVal);
HRESULT ChatMessageType([in] BSTR MessageId, [out, retval] TChatMessageType *pVal);
HRESULT ChatMessageStatus([in] BSTR MessageId, [out, retval] TChatMessageStatus *pVal);
HRESULT ChatMessageLeaveReason([in] BSTR MessageId, [out, retval] TChatLeaveReason *pVal);
HRESULT ChatMessageBody([in] BSTR MessageId, [out, retval] BSTR *pVal);
HRESULT ChatMessageChatname([in] BSTR MessageId, [out, retval] BSTR *pVal);
HRESULT ChatMessageUsers([in] BSTR MessageId, [out, retval] IUserCollection * *pVal);
HRESULT ChatMessageSeen([in] BSTR MessageId);
HRESULT ChatMessageSender([in] BSTR MessageId, [out, retval] IUser* *pVal);
HRESULT ChatMessageIsEditable([in] BSTR MessageId, [out, retval] VARIANT_BOOL* pVal);
HRESULT ChatMessageEditedBy([in] BSTR MessageId, [out, retval] BSTR* pVal);
Val);
01827       [propget, id(14), helpstring(hsIChatMessage_EditTime), helpcontext(47)] HRESULT ChatMessageEditedTimestamp(BSTR MessageId, [out, retval] DATE* pVal);
01828       [propput, id(7), helpstring(hsIChatMessage_Body), helpcontext(47)] HRESULT ChatMessageBody([in] BSTR MessageId, [in] BSTR newVal);
01829       [propget, id(15), helpstring(hsIChatMessage_Role), helpcontext(47)] HRESULT ChatMessageRole([in] BSTR MessageId, [out, retval] TChatMemberRole * pVal);
01830       [propget, id(16), helpstring(hsIChatMessage_Options), helpcontext(47)] HRESULT ChatMessageOptions([in] BSTR MessageId, [out, retval] LONG* pVal );
01831    };
01832     [object,
01833        uuid(6CF6DBDE-AB7C-4635-96ED-2FF525AB4457),
01834        dual,
01835        nonextensible,
01836        pointer_default(unique),
01837        oleautomation,
01838        helpstring(hsIChatMember),
01839        helpcontext(0),
01840        hidden
01841    ]
01842    interface ISkypeChatMember : IDispatch
01843    {
01844        [propget, id(1), helpstring(hsIChatMember_Id)] HRESULT ChatMemberHandle([in] BSTR MemberId, [out, retval] BSTR *pVal);
01845        [propget, id(2), helpstring(hsIChatMember_Role)] HRESULT ChatMemberRole([in] BSTR MemberId, [out, retval] TChatMemberRole* pVal);
HRESULT ChatMemberRole([in] BSTR MemberId, [in] TChatMemberRole newVal);
HRESULT CanSetChatMemberRoleTo([in] BSTR MemberId, [in] TChatMemberRole Role, [out, retval] VARIANT_BOOL* pVal);
HRESULT IsChatMemberActive([in] BSTR MemberId, [out, retval] VARIANT_BOOL* pVal);
HRESULT ChatMemberChat([in] BSTR MemberId, [out, retval] IChat** pVal);

interface ISkypeConference : IDispatch
{
    [propget, id(1), helpstring(hsIConference_Calls), helpcontext(49)] HRESULT ConferenceCalls([in] BSTR ConfId, [out, retval] ICallCollection** pVal);
    [propget, id(2), helpstring(hsIConference_ActiveCalls), helpcontext(49)] HRESULT ConferenceActiveCalls([in] BSTR ConfId, [out, retval] ICallCollection** pVal);
};
interface ISkypeVoicemail : IDispatch
{
    [propget, id(1), helpstring(hsIVoicemail_Type), helpcontext(50)] HRESULT VoicemailType(
        [in] BSTR VoicemailId, [out, retval] TVoicemailType *pVal);
    [propget, id(2), helpstring(hsIVoicemail_PartHandle), helpcontext(50)] HRESULT VoicemailPartHandle(
        [in] BSTR VoicemailId, [out, retval] BSTR *pVal);
    [propget, id(3), helpstring(hsIVoicemail_PartDispName), helpcontext(50)] HRESULT VoicemailPartnerDisplayName(
        [in] BSTR VoicemailId, [out, retval] BSTR *pVal);
    [propget, id(4), helpstring(hsIVoicemail_Status), helpcontext(50)] HRESULT VoicemailStatus(
        [in] BSTR VoicemailId, [out, retval] TVoicemailStatus *pVal);
    [propget, id(5), helpstring(hsIVoicemail_FailureReason), helpcontext(50)] HRESULT VoicemailFailureReason(
        [in] BSTR VoicemailId, [out, retval] TVoicemailFailureReason *pVal);
    [propget, id(6), helpstring(hsIVoicemail_Timestamp), helpcontext(50)] HRESULT VoicemailTimestamp(
        [in] BSTR VoicemailId, [out, retval] DATE *pVal);
01887  [propget, id(7), helpstring(hsIVoicemail_Duration), helpcontext(50)] HRESULT VoicemailDuration([in] BSTR VoicemailId, [out, retval] LONG * pVal);
01888  [propget, id(8), helpstring(hsIVoicemail_AllowedDuration), helpcontext(50)] HRESULT VoicemailAllowedDuration([in] BSTR VoicemailId, [out, retval] LONG * pVal);
01889  [id(9), helpstring(hsIVoicemail_Open), helpcontext(50)] HRESULT OpenVoicemail([in] BSTR VoicemailId);
01890  [id(10), helpstring(hsIVoicemail_StartPlay), helpcontext(50)] HRESULT StartVoicemailPlayback([in] BSTR VoicemailId);
01891  [id(11), helpstring(hsIVoicemail_StopPlay), helpcontext(50)] HRESULT StopVoicemailPlayback([in] BSTR VoicemailId);
01892  [id(12), helpstring(hsIVoicemail_Upload), helpcontext(50)] HRESULT UploadVoicemail([in] BSTR VoicemailId);
01893  [id(13), helpstring(hsIVoicemail_Download), helpcontext(50)] HRESULT DownloadVoicemail([in] BSTR VoicemailId);
01894  [id(14), helpstring(hsIVoicemail_StartRec), helpcontext(50)] HRESULT StartVoicemailRecording([in] BSTR VoicemailId);
01895  [id(15), helpstring(hsIVoicemail_StopRec), helpcontext(50)] HRESULT StopVoicemailRecording([in] BSTR VoicemailId);
01896  [id(16), helpstring(hsIVoicemail_Delete), helpcontext(50)] HRESULT DeleteVoicemail([in] BSTR VoicemailId);
01897  [id(17), helpstring(hsIVoicemail_PlayInCall), helpcontext(50)] HRESULT StartVoicemailPlaybackInCall([in] BSTR VoicemailId);
01898  [id(18), helpstring(hsIVoicemail_Unpla
HRESULT SetVoicemailUnplayed([in] BSTR VoicemailId);

object,
uuid(55A5200C-B2E8-4795-A6FA-858CA3FE2F26),
dual,
nonextensible,
pointer_default(unique),
oleautomation,
helpstring(hsISkypeApplication),
helpcontext(3),
hidden

interface ISkypeApplication : IDispatch
{
    [id(1), helpstring(hsIApp_Create), helpcontext(3)] HRESULT CreateApplication([in] BSTR Name);
    [id(2), helpstring(hsIApp_Delete), helpcontext(3)] HRESULT DeleteApplication([in] BSTR Name);
    [propget, id(3), helpstring(hsIApp_Streams), helpcontext(3)] HRESULT ApplicationStreams([in] BSTR AppName, [out, retval] IApplicationStreamCollection **pVal);
    [id(4), helpstring(hsIApp_Connect), helpcontext(3)] HRESULT ConnectApplicationUser([in] BSTR AppName, [in] BSTR Username, [in, defaultvalue(0)] VARIANT_BOOL WaitConnected);
    [id(5), helpstring(hsIAppStm_Read), helpcontext(3)] HRESULT ReadApplicationStream([in] BSTR AppName, [in] BSTR StreamHandle, [out, retval] BSTR* pVal);
    [id(6), helpstring(hsIAppStm_Write), helpcontext(3)] HRESULT WriteApplicationStream([in] BSTR AppName, [in] BSTR StreamHandle, [out, retval] BSTR* pVal);
}
HRESULT ApplicationConnectableUsers([in] BSTRAppName, [out, retval] IUserCollection** pVal);

HRESULT ApplicationConnectingUsers([in] BSTRAppName, [out, retval] IUserCollection** pVal);

HRESULT DisconnectApplicationUser([in] BSTRAppName, [in] BSTRUsername);

HRESULT SendApplicationDatagram([in] BSTRAppName, [in] BSTRStreamHandle, [in] BSTR Text);

HRESULT ApplicationSendingStreams([in] BSTRAppName, [out, retval] IApplicationStreamCollection** pVal);

HRESULT ApplicationReceivedStreams([in] BSTRAppName, [out, retval] IApplicationStreamCollection** pVal);

interface ISkypeFileTransfer : IDispatch
01936    {
01937        [propget, id(1), helpstring(hsIFileTransfer_Type), helpcontext(0)] HRESULT FileTransferType([in] BSTR TransferId, [out, retval] TFileTransferType* pVal);
01938        [propget, id(2), helpstring(hsIFileTransfer_Status), helpcontext(0)] HRESULT FileTransferStatus([in] BSTR TransferId, [out, retval] TFileTransferStatus* pVal);
01939        [propget, id(3), helpstring(hsIFileTransfer_FailReason), helpcontext(0)] HRESULT FileTransferFailureReason([in] BSTR TransferId, [out, retval] TFileTransferFailureReason* pVal);
01940        [propget, id(4), helpstring(hsIFileTransfer_PartnerHandle), helpcontext(0)] HRESULT FileTransferPartnerHandle([in] BSTR TransferId, [out, retval] BSTR* pVal);
01941        [propget, id(5), helpstring(hsIFileTransfer_PartnerDispName), helpcontext(0)] HRESULT FileTransferPartnerDisplayName([in] BSTR TransferId, [out, retval] BSTR* pVal);
01942        [propget, id(6), helpstring(hsIFileTransfer_StartTime), helpcontext(0)] HRESULT FileTransferStartTime([in] BSTR TransferId, [out, retval] DATE* pVal);
01943        [propget, id(7), helpstring(hsIFileTransfer_FinishTime), helpcontext(0)] HRESULT FileTransferFinishTime([in] BSTR TransferId, [out, retval] DATE* pVal);
01944        [propget, id(8), helpstring(hsIFileTransfer_FilePath), helpcontext(0)] HRESULT FileTransferFilePath([in] BSTR TransferId, [out, retval] BSTR* pVal);
01945        [propget, id(9), helpstring(hsIFileTransfer_FileName), helpcontext(0)] HRESULT FileTransferFileName([in] BSTR TransferId, [out, retval] BSTR* pVal);}
HRESULT FileTransferBytesPerSecond([in] BSTR TransferId, [out, retval] BSTR* pVal);

HRESULT FileTransferBytesTransferred([in] BSTR TransferId, [out, retval] BSTR* pVal);

#endif /*DOXYGEN_SHOULD_SKIP_THIS*/

[
    uuid(F4F90CDD-C620-4118-945E-CAA1BBEBA435),
    helpstring(hsISkypeEvents),
    helpcontext(4)
]

dispinterface _ISkypeEvents
{
    properties:
    methods:
        [id(1), helpstring(hsISkypeEvents_Command),helpcontext(4)] HRESULT Command([in] ICommand * pCommand);
        [id(2), helpstring(hsISkypeEvents_Reply),helpcontext(4)] HRESULT Reply([in] ICommand* pCommand);
        [id(3), helpstring(hsISkypeEvents_Error),helpcontext(4)] HRESULT Error([in] ICommand* pCommand, [in] LONG Number, [in] BSTR Description);
        [id(4), helpstring(hsISkypeEvents_AttachStat),helpcontext(4)] HRESULT AttachmentStatus([in] TAttachmentStatus Status);
        [id(5), helpstring(hsISkypeEvents_ConnStat),helpcontext(4)] HRESULT ConnectionStatus([in] TConnectionStatus Status);
        [id(6), helpstring(hsISkypeEvents_UserStat),helpcontext(4)] HRESULT UserStatus([in] TUse
HRESULT OnlineStatus([in] IUser* pUser, [in] TOnlineStatus Status);
HRESULT CallStatus([in] ICall* pCall, [in] TCallStatus Status);
HRESULT CallHistory();
HRESULT Mute([in] VARIANT_BOOL Mute);
HRESULT MessageStatus([in] IChatMessage* pMessage, [in] TChatMessageStatus Status);
HRESULT MessageHistory([in] BSTR Username);
HRESULT AutoAway([in] VARIANT_BOOL Automatic);
HRESULT CallDtmfReceived([in] ICall* pCall, [in] BSTR Code);
HRESULT VoicemailStatus([in] IVoicemail* pMail, [in] TVoicemailStatus Status);
HRESULT ApplicationConnecting([in] IApplication* pApp, [in] IUserCollection* pUsers);
HRESULT ApplicationStreams([in] IApplication* pApp, [in] IApplicationStreamCollection* pStreams);
01976  [id(18), helpstring(hsISkypeEvents_AppDatagram), helpcontext(4)] HRESULT ApplicationDatagram([in] IApplication* pApp, [in] IApplicationStream* pStream, [in] BSTR Text);
01977  [id(19), helpstring(hsISkypeEvents_AppSending), helpcontext(4)] HRESULT ApplicationSending([in] IApplication* pApp, [in] IApplicationStreamCollection* pStreams);
01978  [id(20), helpstring(hsISkypeEvents_AppReceiving), helpcontext(4)] HRESULT ApplicationReceiving([in] IApplication* pApp, [in] IApplicationStreamCollection* pStreams);
01979  [id(21), helpstring(hsISkypeEvents_ContactFocus), helpcontext(4)] HRESULT ContactsFocused([in] BSTR Username);
01980  [id(22), helpstring(hsISkypeEvents_GroupVisible), helpcontext(4)] HRESULT GroupVisible([in] IGroup* pGroup, [in] VARIANT_BOOL Visible);
01981  [id(23), helpstring(hsISkypeEvents_GroupExpanded), helpcontext(4)] HRESULT GroupExpanded([in] IGroup* pGroup, [in] VARIANT_BOOL Expanded);
01982  [id(24), helpstring(hsISkypeEvents_GroupUsers), helpcontext(4)] HRESULT GroupUsers([in] IGroup* pGroup, [in] IUserCollection* pUsers);
01983  [id(25), helpstring(hsISkypeEvents_GroupDeleted), helpcontext(4)] HRESULT GroupDeleted([in] LONG GroupId);
01984  [id(26), helpstring(hsISkypeEvents_UserMood), helpcontext(4)] HRESULT UserMood([in] IUser* pUser, [in] BSTR MoodText);
01985  [id(27), helpstring(hsISkypeEvents_SmsMessageStatus), helpcontext(4)] HRESULT SmsMessageStatusChanged([in] ISmsMessage* pMessage, [in] TSmsMessageStatus Status);
01986  [id(28), helpstring(hsISkypeEvents_SmsTargetStatus), helpcontext(4)] HRESULT SmsTargetStatusChanged([in] ISmsTarget* pTarget, [in] TSmsTarget Status);
etStatus Status);
01987 [id(29), helpstring(hsISkypeEvents_VoiceInputStatusChanged),helpcontext(4)] HRESULT CallInputStatusChanged([in] ICall* pCall, [in] VARIANT_BOOL Status);
01988 [id(30), helpstring(hsISkypeEvents_AsyncSearchFinished),helpcontext(4)] HRESULT AsyncSearchUsersFinished([in] LONG Cookie, [in] IUserCollection* pUsers);
01989 [id(31), helpstring(hsISkypeEvents_CallSeenStatusChanged),helpcontext(4)] HRESULT CallSeenStatusChanged([in] ICall* pCall, VARIANT_BOOL Status);
01990 [id(32), helpstring(hsISkypeEvents_PluginEventClicked),helpcontext(4)] HRESULT PluginEventClicked([in] IPluginEvent* pEvent);
01991 [id(33), helpstring(hsISkypeEvents_PluginMenuItemClicked),helpcontext(4)] HRESULT PluginMenuItemClicked([in] IPluginMenuItem* pMenuItem, [in] IUserCollection* pUsers, [in] TPluginContext PluginContext, [in] BSTR ContextId);
01992 [id(34), helpstring(hsISkypeEvents_WallpaperChanged),helpcontext(4)] HRESULT WallpaperChanged([in] BSTR Path);
01993 [id(35), helpstring(hsISkypeEvents_FileTransferStatusChanged),helpcontext(4)] HRESULT FileTransferStatusChanged([in] IFileTransfer* pTransfer, [in] TFileTransferStatus Status);
01994 [id(36), helpstring(hsISkypeEvents_CallTransferStatusChanged),helpcontext(4)] HRESULT CallTransferStatusChanged([in] ICall* pCall, [in] TCallStatus Status);
01995 [id(37), helpstring(hsISkypeEvents_ChatMembersChanged),helpcontext(4)] HRESULT ChatMembersChanged([in] IChat* pChat, [in] IUserCollection* pMembers);
01996 [id(38), helpstring(hsISkypeEvents_Cha
tMemberRoleChanged), helpcontext(4)]] HRESULT ChatMemberRoleChanged([in] IChatMember* pMember, [in] TChatMemberRole Role);
01997     [id(39), helpstring(""), helpcontext(4)] HRESULT CallVideoStatusChanged([in] ICall* pCall, [in] TCallVideoStatus Status);
01998     [id(40), helpstring(""), helpcontext(4)] HRESULT CallVideoSendStatusChanged([in] ICall* pCall, [in] TCallVideoSendStatus Status);
01999     [id(41), helpstring(""), helpcontext(4)] HRESULT CallVideoReceiveStatusChanged([in] ICall* pCall, [in] TCallVideoSendStatus Status);
02000     [id(42), helpstring(""), helpcontext(4)] HRESULT SilentModeStatusChanged([in] VARIANT_BOOL Silent);
02001     [id(43), helpstring(""), helpcontext(4)] HRESULT UILanguageChanged([in] BSTR Code);
02002     [id(44), helpstring(""), helpcontext(4)] HRESULT UserAuthorizationRequestReceived([in] IUser* pUser);
02003     
02004 #ifndef DOXYGEN_SHOULD_SKIP_THIS
02005     [
02006       object, 
02007       uuid(3F6758D2-1D3C-4A8F-BD99-6FC6B0E2DC8F), 
02008       dual, 
02009       nonextensible, 
02010       pointer_default(unique), 
02011       oleautomation, 
02012       helpstring(hsISkypeChat), 
02013       helpcontext(48), 
02014       hidden 
02015     ]
02016     interface ISkypeChat : IDispatch 
02017     { 
02018       [propget, id(1), helpstring(hsIChat_Ti
HRESULT ChatTimestamp([in] BSTR ChatName, [out, retval] DATE *pVal);
HRESULT ChatAdder([in] BSTR ChatName, [out, retval] IUser **pVal);
HRESULT ChatStatus([in] BSTR ChatName, [out, retval] TChatStatus *pVal);
HRESULT ChatPosters([in] BSTR ChatName, [out, retval] IUserCollection **pVal);
HRESULT ChatMembers([in] BSTR ChatName, [out, retval] IUserCollection **pVal);
HRESULT ChatTopic([in] BSTR ChatName, [out, retval] BSTR *pVal);
HRESULT ChatTopic([in] BSTR ChatName, [in] BSTR newVal);
HRESULT ChatActiveMembers([in] BSTR ChatName, [out, retval] IUserCollection **pVal);
HRESULT ChatFriendlyName([in] BSTR ChatName, [out, retval] BSTR *pVal);
HRESULT ChatMessages([in] BSTR ChatName, [out, retval] IChatMessageCollection **pVal);
HRESULT OpenChatWindow([in] BSTR ChatName);
HRESULT SendChatMessage([in] BSTR ChatName, [in] BSTR MessageText, [out, retval] IChatMessage **pVal);
HRESULT LeaveChat([in] BSTR ChatName);
HRESULT AddChatMembers([in] BSTR ChatName, [in] IUserCollection* pMembers);
HRESULT ChatRecentMessages([in] BSTR ChatName, [out, retval] IChatMessageCollection **pVal);
HRESULT ChatBookmarked([in] BSTR ChatName, [out, retval] VARIANT_BOOL * pVal);
HRESULT BookmarkChat([in] BSTR ChatName);
HRESULT UnbookmarkChat([in] BSTR ChatName);
HRESULT ChatTopicXML([in] BSTR ChatName, [out, retval] BSTR *pVal);
02040  [propget, id(21), helpstring(hsIChat_Options), helpcontext(48)] HRESULT ChatOptions([in] BSTR ChatName, [out, retval] LONG* pVal);
02041  [propput, id(21), helpstring(hsIChat_Options), helpcontext(48)] HRESULT ChatOptions([in] BSTR ChatName, [in] LONG newVal);
02042  [propget, id(22), helpstring(hsIChat_Passwordhint), helpcontext(48)] HRESULT ChatPasswordHint([in] BSTR ChatName, [out, retval] BSTR* pVal);
02043  [id(23), helpstring(hsIChat_Password), helpcontext(48)] HRESULT SetChatPassword([in] BSTR ChatName, [in] BSTR Password, [in, defaultvalue("")]] BSTR Hint);
02044  [propget, id(24), helpstring(hsIChat_Guidelines), helpcontext(48)] HRESULT ChatGuideLine(s([in] BSTR ChatName, [out, retval] BSTR* pVal);
02045  [propput, id(24), helpstring(hsIChat_Guidelines), helpcontext(48)] HRESULT ChatGuideLines([in] BSTR ChatName, [in] BSTR newVal);
02046  [propget, id(25), helpstring(hsIChat_Description), helpcontext(48)] HRESULT ChatDescription([in] BSTR ChatName, [out, retval] BSTR* pVal);
02047  [propput, id(25), helpstring(hsIChat_Description), helpcontext(48)] HRESULT ChatDescription([in] BSTR ChatName, [in] BSTR newVal);
02048  [propget, id(26), helpstring(hsIChat_DialogPartner), helpcontext(48)] HRESULT ChatDialogPartner([in] BSTR ChatName, [out, retval] BSTR* pVal);
02049  [propget, id(27), helpstring(hsIChat_ActiveTime), helpcontext(48)] HRESULT ChatActivityTimestamp([in] BSTR ChatName, [out, retval] DATE* pVal);
02050  [propget, id(28), helpstring(hsIChat_MyRole), helpcontext(48)] HRESULT ChatMyRole([in] BSTR ChatName, [out, retval] TChatMemberRole* pVal) ;
HRESULT ChatApplicants([in] BSTR ChatName, [out, retval] IUserCollection ** pVal);
HRESULT JoinChat([in] BSTR ChatName);
HRESULT KickChatMember([in] BSTR ChatName, [in] BSTR Handle);
HRESULT KickBanChatMember([in] BSTR ChatName, [in] BSTR Handle);
HRESULT DisbandChat([in] BSTR ChatName);
HRESULT EnterChatPassword([in] BSTR ChatName, [in] BSTR Password);
HRESULT ClearChatRecentMessages([in] BSTR ChatName);
HRESULT AcceptSharedGroupAdd([in] BSTR ChatName);
HRESULT ChatAlertString([in] BSTR ChatName, [in] BSTR newVal);
HRESULT ChatType([in] BSTR ChatName, [out, retval] TChatType* pVal);
HRESULT ChatMyStatus([in] BSTR ChatName, [out, retval] TChatMyStatus* pVal);
interface ISkypeGroup : IDispatch
{
    [propget, id(1), helpstring(hsISkypeGroup_Type), helpcontext(9)] HRESULT GroupType([in] BSTR GroupId, [out, retval] TGroupType *pVal);
    [propget, id(2), helpstring(hsISkypeGroup_CustomId), helpcontext(9)] HRESULT GroupCustomId([in] BSTR GroupId, [out, retval] BSTR *pVal);
    [propget, id(3), helpstring(hsISkypeGroup_DisplayName), helpcontext(9)] HRESULT GroupDisplayName([in] BSTR GroupId, [out, retval] BSTR *pVal);
    [propput, id(3), helpstring(hsISkypeGroup_DisplayName), helpcontext(9)] HRESULT GroupDisplayName([in] BSTR GroupId, [in] BSTR newVal);
    [propget, id(4), helpstring(hsISkypeGroup_Users), helpcontext(9)] HRESULT GroupUsers([in] BSTR GroupId, [out, retval] IUserCollection* *pVal);
    [propget, id(5), helpstring(hsISkypeGroup_IsVisible), helpcontext(9)] HRESULT GroupIsVisible([in] BSTR GroupId, [out, retval] VARIANT_BOOL *pVal);
    [propget, id(6), helpstring(hsISkypeGroup_IsExpanded), helpcontext(9)] HRESULT GroupIsExpanded([in] BSTR GroupId, [out, retval] VARIANT_BOOL *pVal);
    [id(7), helpstring(hsISkypeGroup_AddUser),
HRESULT AddGroupUser([in] BSTR GroupId, [in] BSTR Username);
HRESULT RemoveGroupUser([in] BSTR GroupId, [in] BSTR Username);
HRESULT ShareGroup([in] BSTR GroupId, [in] BSTR MessageText);
HRESULT AcceptSharedGroup([in] BSTR GroupId);
HRESULT DeclineSharedGroup([in] BSTR GroupId);

interface ISkypeSettings : IDispatch
{  
    [propget, id(1), helpstring(hsISettings_AudioIn), helpcontext(2)] HRESULT AudioIn([out, retval] BSTR *pVal);
    [propput, id(1), helpstring(hsISettings_AudioIn), helpcontext(2)] HRESULT AudioIn([in] BSTR newVal);
    [propget, id(2), helpstring(hsISettings_AudioOut), helpcontext(2)] HRESULT AudioOut([out, retval] BSTR *pVal);
02105 [propget, id(2), helpstring(hsISetting s_AudioOut), helpcontext(2)] HRESULT AudioOut([in] BSTR newVal);
02106 [propget, id(3), helpstring(hsISetting s_AutoAway), helpcontext(2)] HRESULT AutoAway([out, retval] VARIANT_BOOL *pVal);
02107 [propget, id(3), helpstring(hsISetting s_AutoAway), helpcontext(2)] HRESULT AutoAway([in] VARIANT_BOOL newVal);
02108 [propget, id(4), helpstring(hsISetting s_Ringer), helpcontext(2)] HRESULT Ringer([out, retval] BSTR *pVal);
02109 [propget, id(4), helpstring(hsISetting s_Ringer), helpcontext(2)] HRESULT Ringer([in] BSTR newVal);
02110 [propget, id(5), helpstring(hsISetting s_Avatar), helpcontext(2)] HRESULT Avatar([in, defaultvalue("1"), BSTR Id, [in] BSTR newVal);
02111 [propget, id(6), helpstring(hsISetting s_RingToneStatus), helpcontext(2)] HRESULT RingToneStatus([in, defaultvalue("1"), BSTR Id, [out, retval] VARIANT_BOOL *pVal);
02112 [propget, id(7), helpstring(hsISetting s_RingTone), helpcontext(2)] HRESULT RingTone([in, defaultvalue("1"), BSTR Id, [in] BSTR newVal);
02113 [propget, id(8), helpstring(hsISetting s_VideoIn), helpcontext(2)] HRESULT VideoIn([out, retval] BSTR *pVal);
02114 [propget, id(8), helpstring(hsISetting s_VideoIn), helpcontext(2)] HRESULT VideoIn([in] BSTR newVal);
02115 [propget, id(9), helpstring(hsISetting s_PCSpeaker), helpcontext(2)] HRESULT PCSpeaker([out, retval] VARIANT_BOOL *pVal);
02116 [propget, id(9), helpstring(hsISetting s_PCSpeaker), helpcontext(2)] HRESULT PCSpeaker([in] VARIANT_BOOL newVal);
HRESULT AGC([out, retval] VARIANT_BOOL *pVal);

HRESULT AEC([in] VARIANT_BOOL newVal);

HRESULT Language([out, retval] BSTR *pVal);

HRESULT AEC([in] VARIANT_BOOL newVal);

HRESULT Language([in] BSTR newVal);

HRESULT RingToneStatus([in, defaultvalue("1")]) BSTR Id, [in] VARIANT_BOOL newVal);

HRESULT SaveAvatarToFile([in] BSTR Filename, [in, defaultvalue("1")]) BSTR AvatarId);

HRESULT LoadAvatarFromFile([in] BSTR Filename, [in, defaultvalue("1")]) BSTR AvatarId);

HRESULT ResetIdleTimer(void);

object,
    uuid(B6BED647-709E-4B4B-B1E3-B0105999966F),
dual,
nonextensible,
pointer_default(unique),
oleautomation,
helpstring(hsISkypeClient),
helpcontext(36),
hidden
]
interface ISkypeClient : IDispatch
{
    [id(1), helpstring(hsIClient_Start), helpcontext(36)] HRESULT Start([in, defaultvalue(0)] VARIANT_BOOL Minimized, [in, defaultvalue(0)] VARIANT_BOOL Nosplash);
    [id(2), helpstring(hsIClient_Minimize), helpcontext(36)] HRESULT Minimize();
    [id(3), helpstring(hsIClient_Shutdown), helpcontext(36)] HRESULT Shutdown();
    [propget, id(4), helpstring(hsIClient_Running), helpcontext(36)] HRESULT IsRunning([out, retval] VARIANT_BOOL *pVal);
    [id(5), helpstring(hsIClient_OpenProfile), helpcontext(36)] HRESULT OpenProfileDialog();
    [id(6), helpstring(hsIClient_OpenUserInfo), helpcontext(36)] HRESULT OpenUserInfoDialog([in] BSTR Username);
    [id(7), helpstring(hsIClient_OpenConference), helpcontext(36)] HRESULT OpenConferenceDialog();
    [id(8), helpstring(hsIClient_OpenSearch), helpcontext(36)] HRESULT OpenSearchDialog();
    [id(9), helpstring(hsIClient_OpenOptions), helpcontext(36)] HRESULT OpenOptionsDialog([in] BSTR Page);
    [id(10), helpstring(hsIClient_OpenCallHist), helpcontext(36)] HRESULT OpenCallHistoryTab();
HRESULT OpenContactsTab();
HRESULT OpenDialpadTab();
HRESULT OpenSendContactsDialog([in, defaultvalue(""')] BSTR Username);
HRESULT OpenBlockedUsersDialog();
HRESULT OpenImportContactsWizard();
HRESULT OpenGettingStartedWizard();
HRESULT OpenAuthorizationDialog([in] BSTR Username);
HRESULT OpenDialog([in] BSTR Name, [in, defaultvalue(""')] BSTR Param1, [in, defaultvalue(""')] BSTR Param2);
HRESULT OpenVideoTestDialog();
HRESULT OpenAddContactDialog([in, defaultvalue(""')] BSTR Username);
HRESULT OpenMessageDialog([in] BSTR Username, [in, defaultvalue(""')] BSTR Text);
HRESULT OpenFileTransferDialog([in] BSTR Username, [in] BSTR Folder);
elpcontext(36)] HRESULT Focus();
02164 [id(24), helpstring(hsIClient_BtnPressed), helpcontext(36)] HRESULT ButtonPressed([in] BSTR Key);
02165 [id(25), helpstring(hsIClient_BtnReleased), helpcontext(36)] HRESULT ButtonReleased([in] BSTR Key);
02166 [id(26), helpstring(hsIClient_OpenSms), helpcontext(36)] HRESULT OpenSmsDialog([in] BSTR SmsId);
02167 [propget, id(27), helpstring(hsIClient_Wallpaper), helpcontext(36)] HRESULT Wallpaper([out, retval] BSTR* pVal);
02168 [propput, id(27), helpstring(hsIClient_Wallpaper), helpcontext(36)] HRESULT Wallpaper([in] BSTR newVal);
02169 [id(28), helpstring(hsIClient_OpenLive), helpcontext(36)] HRESULT OpenLiveTab();
02170 }
02171 [object,
02172 uuid(B77A3D1F-6847-402f-BB82-A0564379088E),
02174 dual,
02175 nonextensible,
02176 pointer_default(unique),
02177 oleautomation,
02178 helpstring(""),
02179 helpcontext(0),
02180 hidden
02181 ]
02182 interface ISkypePlugin : IDispatch
02183 {
02184 [id(1), helpstring(""), helpcontext(0)] HRESULT CreatePluginEvent([in] BSTR EventId, [in] BSTR Caption, [in] BSTR Hint, [out, retval] IPlug inEvent** pEvent);
HRESULT DeletePluginEvent([in] BSTR EventId);

HRESULT CreatePluginMenuItem([in] BSTR MenuItemId, [in] TPluginContext PluginContext, [in] BSTR CaptionText, [in,defaultvalue(""')] BSTR HintText, [in, defaultvalue("'')] BSTR IconPath, [in, defaultvalue(-1)] VARIANT_BOOL Enabled, [in, defaultvalue(pluginContactTypeAll)] TPluginContactType ContactType, [in, defaultvalue(0)] VARIANT_BOOL MultipleContacts, [out,retval] IPluginMenuItem** pMenuItem);

HRESULT DeletePluginMenuItem([in] BSTR MenuItemId);

[propput, id(5), helpstring(""'), helpcontext(0)] HRESULT PluginMenuItemCaption([in] BSTR MenuItemId, [in] BSTR newVal);

[propput, id(6), helpstring(""'), helpcontext(0)] HRESULT PluginMenuItemHint([in] BSTR MenuItemId, [in] BSTR newVal);

[id(7), helpstring(""'), helpcontext(0)] HRESULT EnablePluginMenuItem([in] BSTR MenuItemId, [in] VARIANT_BOOL Enabled);

};

#endif /*DOXYGEN_SHOULD_SKIP_THIS*/

interface ICallChannelManager;

[ object,
  uuid(B76C04C9-0EB1-4748-B0C0-D01DEB595906),
  dual,
  nonextensible,
  pointer_default(unique),
  oleautomation,
  helpstring(hsICallChannel),
  helpcontext(67)
interface ICallChannel : IDispatch
{
    [propget, id(2), helpstring(hsICallChannel_Type), helpcontext(67)] HRESULT Type([out, retval] TCallChannelType *pVal);
    [propput, id(2), helpstring("hsICallChannel_Type"), helpcontext(67)] HRESULT Type([in] TCallChannelType newVal);
    [id(3), helpstring(hsICallChannel_Send), helpcontext(67)] HRESULT SendTextMessage(BSTR Text);
    [propget, id(4), helpstring(hsICallChannel_Stream), helpcontext(67)] HRESULT Stream([out, retval] IApplicationStream* *pVal);
    [propget, id(5), helpstring(hsICallChannel_Manager), helpcontext(67)] HRESULT Manager([out, retval] ICallChannelManager* *pVal);
    [propget, id(6), helpstring(hsICallChannel_Call), helpcontext(67)] HRESULT Call([out, retval] ICall* *pVal);
};

[object, uuid(8CB09087-E389-4C6D-A6A2-7D4BCA85D9), dual, nonextensible, pointer_default(unique), oleautomation, helpstring(hsICallChannelCollection), helpcontext(68)]

interface ICallChannelCollection : IDispatch
{
    [propget, id(1), helpstring(hsICallCh
HRESULT Count([out, retval] LONG* pVal);

HRESULT Add([in] ICallChannelColl* pChannel);

HRESULT Remove([in] long Index);

HRESULT RemoveAll(void);

HRESULT Item([in] LONG Index, [out, retval] ICallChannel** pVal);

HRESULT _NewEnum([out, retval] IUnknown** pVal);

interface ICallChannelManager : IDispatch
{
    [id(1), helpstring(hsICallChannelMgr_Connect), helpcontext(69)] HRESULT Connect([in] ISkype* pSkype);
    [id(2), helpstring(hsICallChannelMgr_Disconnect), helpcontext(69)] HRESULT Disconnect();
}
HRESULT Channels([out, retval] ICallChannelCollection** pVal);

HRESULT ChannelType([out, retval] TCallChannelType *pVal);

HRESULT ChannelType([in] TCallChannelType newVal);

HRESULT Name([out, retval] BSTR *pVal);

HRESULT Name([in] BSTR newVal);

HRESULT CreateApplication([in, defaultvalue("")]} BSTR ApplicationName);

HRESULT Created([out, retval] VARIANT_BOOL *pVal);

};

[object,
 uuid(0EF6FD5D-607D-4EA3-9C90-683D73449E9D),
 dual,
 nonextensible,
 pointer_default(unique),
 oleautomation,
 helpstring(hsICallChannelMsg),
 helpcontext(70)
]

interface ICallChannelMessage : IDispatch
{
    [propget, id(1), helpstring(hsICallChannelMsg_Text), helpcontext(70)] HRESULT Text([out,
retval] BSTR *pVal);  
02268 [propput, id(1), helpstring(hsICallChannelMsg_Text), helpcontext(70)] HRESULT Text([in] BSTR newVal);
02269 }
02270 [uuid(497ABB45-20AE-49D1-A39D-CBE84A31B20C),
02271 helpstring(ICallChannelManagerEvents),
02272 helpcontext(71)
02273 ]
02274 dispinterface _ICallChannelManagerEvents
02275 {
02276 properties:
02277 methods:
02278 [id(1), helpstring(ICallChannelManagerEvents_Channels),helpcontext(71)] HRESULT Channels ([in] ICallChannelManager* pManager, [in] ICallChannelCollection* pChannels);
02279 [id(2), helpstring(ICallChannelManagerEvents_Message),helpcontext(71)] HRESULT Message([in] ICallChannelManager* pManager, [in] ICallChannel* pChannel, [in] ICallChannelMessage* pMessage);
02280 [id(3), helpstring(ICallChannelManagerEvents_Created),helpcontext(71)] HRESULT Created();
02281 }
02282 [uuid(03282B5D-B38F-469D-849A-09B0A7F4881B),
02283 version(1.0),
02284 helpstring(hsSkypeLib_Version), helpcontext(1),
02285 helpfile(hsSkypeLib_HelpFile)
02286 ]
02287 library SKYPE4COMLib
02288 {

importlib("stdole32.tlb");
importlib("stdole2.tlb");
interface IApplication;
interface IApplicationStream;
interface IApplicationStreamCollection;
interface ICall;
interface ICallChannel;
interface ICallChannelManager;
interface ICallChannelMessage;
interface ICallCollection;
interface ICallIoDevice;
interface ICallIoDeviceCollection;
interface IChat;
interface IChatCollection;
interface IChatMessage;
interface IChatMessageCollection;
interface IChatMember;
interface IChatMemberCollection;
interface IClient;
interface ICommand;
interface IConference;
interface IConferenceCollection;
interface IConversion;
interface IFileTransfer;
interface IFileTransferCollection;
interface IParticipant;
interface IParticipantCollection;
interface ISettings;
interface ISkype;
interface ISkypeApplication;
interface ISkypeCall;
interface ISkypeChat;
interface ISkypeChatMessage;
interface ISkypeChatMember;
interface ISkypeConference;
interface ISkypeDevice;
interface ISkypeFileTransfer;
interface ISkypeSms;
interface ISkypeUser;
interface ISkypeVoicemail;
interface IUser;
interface IUserCollection;
interface IVoicemail;
interface IVoicemailCollection;
dispinterface _ISkypeEvents;
dispinterface _ICallChannelManagerEvents;

 coclass Skype
{
[default] interface ISkype;
[default, source] dispinterface _ISkypeEvents;
interface ISkypeApplication;
interface ISkypeCall;
interface ISkypeChat;
interface ISkypeChatMessage;
interface ISkypeConference;
interface ISkypeFileTransfer;
interface ISkypePlugin;
interface ISkypeSms;
interface ISkypeUser;
interface ISkypeVoicemail;
};

class User
```csharp
{ [default] interface IUser;
};
[
uuid(7B030003-037D-490D-9169-A4F391B3D831),
helpstring(hsUserCollection), helpcontext(20)
] coclass UserCollection
{
[default] interface IUserCollection;
};
[
uuid(2EEAB6D0-491E-4962-BBA1-FF1CCA6D4DD0),
helpstring(hsConversion), helpcontext(21)
] coclass Conversion
{
[default] interface IConversion;
};
[
uuid(D26B1D42-9C42-4E7B-BB73-86384C4B4345),
helpstring(hsCall), helpcontext(22)
] coclass Call
{
[default] interface ICall;
};
[
uuid(F3188CF3-EF22-4C5B-92CB-605964761C3B),
helpstring(hsCallCollection), helpcontext(23)
}
coclass CallCollection
{
    [default] interface ICallCollection;
}

[uuid(7ACDC5B4-76A1-4BDF-918D-6962FCABBAD3),
     helpstring(hsChat), helpcontext(34)
]
coclass Chat
{
    [default] interface IChat;
}

[uuid(15B6FEE5-5FB3-4071-AC1F-7AEDC0E2A6BB),
     helpstring(hsChatColl), helpcontext(35)
]
coclass ChatCollection
{
    [default] interface IChatCollection;
}

[uuid(52071016-E648-4D3B-B57E-2B46CC993CE0),
     helpstring(hsParticipant), helpcontext(41)
]
coclass Participant
{
    [default] interface IParticipant;
}

[uuid(E1BC9147-C3E3-4E8A-8304-5E6B5C1C0774),
 coclass ParticipantCollection
{ [default] interface IParticipantCollection;
  };
 [ uuid(A983C9EC-D73E-4364-B89B-ACD1E405674F),
  helpstring(hsConference), helpcontext(43)
  ]}

coclass Conference
{ [default] interface IConference;
  };
 [ uuid(3506CDB7-8BC6-40C0-B108-CEA0B9480130),
  helpstring(hsConferenceColl), helpcontext(44)
  ]}

coclass ConferenceCollection
{ [default] interface IConferenceCollection;
  };
 [ uuid(DD0E8ED5-1494-4B87-A35C-39F6ED4B1153),
  helpstring(hsVoicemail), helpcontext(53)
  ]}

coclass Voicemail
{
[default] interface IVoicemail;
}
[
    uuid(A7DF2611-D752-4C9F-A90A-B56F18485EE9),
    helpstring(hsVoicemailColl), helpcontext(54)
]
coclass VoicemailCollection
{
    [default] interface IVoicemailCollection;
};
[
    uuid(29DCD339-D184-469B-8BFB-199A2CCF014E),
    helpstring(hsApplication), helpcontext(29)
]
coclass Application
{
    [default] interface IApplication;
};
[
    uuid(6FA10A39-4760-4C94-A210-2398848618EC),
    helpstring(hsApplicationStream), helpcontext(55)
]
coclass ApplicationStream
{
    [default] interface IApplicationStream;
}
[
    uuid(4B42750B-57A1-47E7-B340-8EAE0E3126A4),
helpstring(hsApplicationStreamColl), helpcontext(57)

class ApplicationStreamCollection
{
    [default] interface IApplicationStreamCollection;
}

uuid(9017071A-2E34-4C3A-9BBB-688CBB5A9FF2),
helpstring(hsChatMessage), helpcontext(24)

class ChatMessage
{
    [default] interface IChatMessage;
}

uuid(10DD084E-A5AE-456F-A3BE-DA67EBE6B090),
helpstring(hsChatMessageColl), helpcontext(25)

class ChatMessageCollection
{
    [default] interface IChatMessageCollection;
}

uuid(452CCB69-6A95-4370-9E5A-B3EFB06A761),
helpstring(hsProfile), helpcontext(61)

class Profile
{
    [default] interface IProfile;
coclass Group {
    [default] interface IGroup;
};

coclass GroupCollection {
    [default] interface IGroupCollection;
};

coclass Settings {
    [default] interface ISettings;
};

coclass Client {

[default] interface IClient;

uuid(2DBCDA9F-1248-400B-A382-A56D71BF7B15),
helpstring(hsCommand), helpcontext(26)
]
coclass Command
{
[default] interface ICommand;
}

uuid(89DD2F9D-C325-48BF-A615-96BD039BB83),
helpstring(hsCallChannel), helpcontext(63)
]
coclass CallChannel
{
[default] interface ICallChannel;
}

uuid(42FE718B-A148-41D6-885B-01A0AFAE8723),
helpstring(hsCallChannelColl), helpcontext(64)
]
coclass CallChannelCollection
{
[default] interface ICallChannelCollection;
}

uuid(5E541E71-A474-4EAD-8FCB-24D400D023B7),
helpstring(hsCallChannelMgr), helpcontext(63)
coclass CallChannelManager
{
    [default] interface ICallChannelManager;
    [default, source] dispinterface _ICallChannelManagerEvents;
}

 coclass CallChannelMessage
{
    [default] interface ICallChannelMessage;
}

ifndef DOXYGEN_SHOULD_SKIP_THIS

 coclass IEProtocolHandler
{
    [default] interface IUnknown;
}

endif /*DOXYGEN_SHOULD_SKIP_THIS*/

 coclass

02595  coclass SmsMessage
02596  {
02597      [default] interface ISmsMessage;
02598  }
02599  [uuid(1BCA4635-F1FC-44C8-B829-48229AEB32E3),
02600      helpstring(hsSmsMessageCollection), helpcontext(80)
02601  ]
02602  coclass SmsMessageCollection
02603  {
02604      [default] interface ISmsMessageCollection;
02605  }
02606  [uuid(5792FC7D-5E1D-4F1A-BD4F-A7A50F92BC6E),
02607      helpstring(hsSmsChunk), helpcontext(83)
02608  ]
02609  coclass SmsChunk
02610  {
02611      [default] interface ISmsChunk;
02612  }
02613  [uuid(CC461FC3-C9BE-41FB-8E47-E0115CBC01CC),
02614      helpstring(hsSmsChunkCollection), helpcontext(84)
02615  ]
02616  coclass SmsChunkCollection
02617  {
02618      [default] interface ISmsChunkCollection;
02619  }
02620  [
coclass SmsTarget
{
    [default] interface ISmsTarget;
}

coclass SmsTargetCollection
{
    [default] interface ISmsTargetCollection;
}

coclass PluginMenuItem
{
    [default] interface IPluginMenuItem;
}

coclass PluginEvent
{ [default] interface IPluginEvent; }

Copyright © 2006 Skype Limited. All rights reserved.
Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
**_ICallChannelManagerEvents Member List**

This is the complete list of members for **_ICallChannelManagerEvents**, including all inherited members.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Channels()</td>
<td>([in] ICallChannelManager *pManager, [in] ICallChannelCollection *pChannels)</td>
</tr>
<tr>
<td>Created()</td>
<td><strong>_ICallChannelManagerEvents</strong></td>
</tr>
<tr>
<td>Message()</td>
<td>([in] ICallChannelManager *pManager, [in] ICallChannel *pChannel, [in] ICallChannelMessage *pMessage)</td>
</tr>
</tbody>
</table>

Copyright © 2006 Skype Limited. All rights reserved.
Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
This is the complete list of members for `_ISkypeEvents`, including all inherited members.

<table>
<thead>
<tr>
<th>Method Name</th>
<th>Parameters</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>ApplicationConnecting</td>
<td>([in] IApplication *pApp,[in] IUserCollection *pUsers)</td>
<td>_ISkypeEvents</td>
</tr>
<tr>
<td>ApplicationDatagram</td>
<td>([in] IApplication *pApp,[in] IApplicationStream *pStream,[in] BSTR Text)</td>
<td>_ISkypeEvents</td>
</tr>
<tr>
<td>ApplicationReceiving</td>
<td>([in] IApplication *pApp,[in] IApplicationStreamCollection *pStreams)</td>
<td>_ISkypeEvents</td>
</tr>
<tr>
<td>ApplicationSending</td>
<td>([in] IApplication *pApp,[in] IApplicationStreamCollection *pStreams)</td>
<td>_ISkypeEvents</td>
</tr>
<tr>
<td>ApplicationStreams</td>
<td>([in] IApplication *pApp,[in] IApplicationStreamCollection *pStreams)</td>
<td>_ISkypeEvents</td>
</tr>
<tr>
<td>AsyncSearchUsersFinished</td>
<td>([in] LONG Cookie,[in] IUserCollection *pUsers)</td>
<td>_ISkypeEvents</td>
</tr>
<tr>
<td>AttachmentStatus</td>
<td>([in] TAttachmentStatus Status)</td>
<td>_ISkypeEvents</td>
</tr>
<tr>
<td>AutoAway</td>
<td>([in] VARIANT_BOOL Automatic)</td>
<td>_ISkypeEvents</td>
</tr>
<tr>
<td>CallDtmfReceived</td>
<td>([in] ICall *pCall,[in] BSTR Code)</td>
<td>_ISkypeEvents</td>
</tr>
<tr>
<td>CallHistory</td>
<td>()</td>
<td>_ISkypeEvents</td>
</tr>
<tr>
<td>CallInputStatusChanged</td>
<td>([in] ICall *pCall,[in] VARIANT_BOOL Status)</td>
<td>_ISkypeEvents</td>
</tr>
<tr>
<td>CallSeenStatusChanged</td>
<td>([in] ICall *pCall, [in] VARIANT_BOOL Status)</td>
<td>_ISkypeEvents</td>
</tr>
<tr>
<td>CallStatus</td>
<td>([in] ICall *pCall,[in] TCallStatus Status)</td>
<td>_ISkypeEvents</td>
</tr>
<tr>
<td>CallTransferStatusChanged</td>
<td>([in] ICall *pCall,[in] TCallStatus Status)</td>
<td>_ISkypeEvents</td>
</tr>
<tr>
<td>CallVideoReceiveStatusChanged</td>
<td>([in] ICall *pCall, [in] TCallVideoSendStatus Status)</td>
<td>_ISkypeEvents</td>
</tr>
<tr>
<td>CallVideoSendStatusChanged</td>
<td>([in] ICall *pCall,[in] TCallVideoSendStatus Status)</td>
<td>_ISkypeEvents</td>
</tr>
<tr>
<td>CallVideoStatusChanged</td>
<td>([in] ICall *pCall,[in] TCallVideoStatus Status)</td>
<td>_ISkypeEvents</td>
</tr>
<tr>
<td>ChatMemberRoleChanged</td>
<td>([in] IChatMember)</td>
<td>_ISkypeEvents</td>
</tr>
<tr>
<td>Event Name</td>
<td>Parameters</td>
<td>Event Class</td>
</tr>
<tr>
<td>------------------------------------------------</td>
<td>----------------------------------------------------------------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>ChatMembersChanged</td>
<td>[in] IChat *pChat, [in] IUserCollection *pMembers</td>
<td>_ISkypeEvents</td>
</tr>
<tr>
<td>Command</td>
<td>[in] ICommand *pCommand</td>
<td>_ISkypeEvents</td>
</tr>
<tr>
<td>ConnectionStatus</td>
<td>[in] TConnectionStatus Status</td>
<td>_ISkypeEvents</td>
</tr>
<tr>
<td>ContactsFocused</td>
<td>[in] BSTR Username</td>
<td>_ISkypeEvents</td>
</tr>
<tr>
<td>GroupDeleted</td>
<td>[in] LONG GroupId</td>
<td>_ISkypeEvents</td>
</tr>
<tr>
<td>MessageHistory</td>
<td>[in] BSTR Username</td>
<td>_ISkypeEvents</td>
</tr>
<tr>
<td>Mute</td>
<td>[in] VARIANT_BOOL Mute</td>
<td>_ISkypeEvents</td>
</tr>
<tr>
<td>OnlineStatus</td>
<td>[in] IUser *pUser, [in] TOnlineStatus Status</td>
<td>_ISkypeEvents</td>
</tr>
<tr>
<td>PluginEventClicked</td>
<td>[in] IPluginEvent *pEvent</td>
<td>_ISkypeEvents</td>
</tr>
<tr>
<td>Reply</td>
<td>[in] ICommand *pCommand</td>
<td>_ISkypeEvents</td>
</tr>
<tr>
<td>SilentModeStatusChanged</td>
<td>[in] VARIANT_BOOL Silent</td>
<td>_ISkypeEvents</td>
</tr>
<tr>
<td>SmsTargetStatusChanged</td>
<td>[in] ISmsTarget *pTarget, [in] TSmsTargetStatus Status</td>
<td>_ISkypeEvents</td>
</tr>
<tr>
<td>UILanguageChanged</td>
<td>[in] BSTR Code</td>
<td>_ISkypeEvents</td>
</tr>
<tr>
<td>Event</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>--------------------------------------------</td>
<td>-------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>UserAuthorizationRequestReceived</td>
<td>([in] IUser *pUser)</td>
<td></td>
</tr>
<tr>
<td>UserMood</td>
<td>([in] IUser *pUser,[in] BSTR MoodText)</td>
<td></td>
</tr>
<tr>
<td>UserStatus</td>
<td>([in] TUserStatus Status)</td>
<td></td>
</tr>
<tr>
<td>VoicemailStatus</td>
<td>([in] IVoicemail *pMail,[in] TVoicemailStatus Status)</td>
<td></td>
</tr>
<tr>
<td>WallpaperChanged</td>
<td>([in] BSTR Path)</td>
<td></td>
</tr>
</tbody>
</table>

*ISkypeEvents*
Skype4COM 1.0.36.0

Main Page  Related Pages  Modules  Classes
Files  Examples
Class List  Class Members
# IApplication Member List

This is the complete list of members for **IApplication**, including all inherited members.

<table>
<thead>
<tr>
<th>Member Name</th>
<th>Description</th>
<th>IInterface</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Connect</strong></td>
<td>(in) BSTR Username, (in, defaultvalue(0)) VARIANT_BOOL WaitConnected</td>
<td>IApplication</td>
</tr>
<tr>
<td><strong>ConnectableUsers</strong></td>
<td></td>
<td>IApplication</td>
</tr>
<tr>
<td><strong>ConnectingUsers</strong></td>
<td></td>
<td>IApplication</td>
</tr>
<tr>
<td><strong>Create()</strong></td>
<td></td>
<td>IApplication</td>
</tr>
<tr>
<td><strong>Delete()</strong></td>
<td></td>
<td>IApplication</td>
</tr>
<tr>
<td><strong>Name</strong></td>
<td></td>
<td>IApplication</td>
</tr>
<tr>
<td><strong>ReceivedStreams</strong></td>
<td></td>
<td>IApplication</td>
</tr>
<tr>
<td><strong>SendDatagram</strong></td>
<td>(in) BSTR Text, (in, defaultvalue(0)) IApplicationStreamCollection *pStreams</td>
<td>IApplication</td>
</tr>
<tr>
<td><strong>SendingStreams</strong></td>
<td></td>
<td>IApplication</td>
</tr>
<tr>
<td><strong>Streams</strong></td>
<td></td>
<td>IApplication</td>
</tr>
</tbody>
</table>

Copyright 🌐 2006 Skype Limited. All rights reserved.
<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td>Class List</td>
<td>Class Members</td>
</tr>
</tbody>
</table>
This is the complete list of members for **IApplicationStream**, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>ApplicationName</td>
<td>IApplicationStream</td>
</tr>
<tr>
<td>DataLength</td>
<td>IApplicationStream</td>
</tr>
<tr>
<td>Disconnect()</td>
<td>IApplicationStream</td>
</tr>
<tr>
<td>Handle</td>
<td>IApplicationStream</td>
</tr>
<tr>
<td>PartnerHandle</td>
<td>IApplicationStream</td>
</tr>
<tr>
<td>Read([out, retval] BSTR *pText)</td>
<td>IApplicationStream</td>
</tr>
<tr>
<td>SendDatagram([in] BSTR Text)</td>
<td>IApplicationStream</td>
</tr>
<tr>
<td>Write([in] BSTR Text)</td>
<td>IApplicationStream</td>
</tr>
</tbody>
</table>

Copyright © 2006 Skype Limited. All rights reserved.
### Skype4COM 1.0.36.0

#### Main Page

#### Related Pages

#### Modules

#### Classes

#### Files

#### Examples

| Class List | Class Members |
IApplicationStreamCollection Member List

This is the complete list of members for
IApplicationStreamCollection, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>_NewEnum</td>
<td>IApplicationStreamCollection</td>
</tr>
<tr>
<td>Add([in] IApplicationStream *pItem)</td>
<td>IApplicationStreamCollection</td>
</tr>
<tr>
<td>Count</td>
<td>IApplicationStreamCollection</td>
</tr>
<tr>
<td>Item</td>
<td>IApplicationStreamCollection</td>
</tr>
<tr>
<td>Remove([in] long Index)</td>
<td>IApplicationStreamCollection</td>
</tr>
<tr>
<td>RemoveAll( void)</td>
<td>IApplicationStreamCollection</td>
</tr>
</tbody>
</table>

Copyright © 2006 Skype Limited. All rights reserved.
<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## ICall Member List

This is the complete list of members for **ICall**, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Answer()</td>
<td>ICall</td>
</tr>
<tr>
<td>CanTransfer</td>
<td>ICall</td>
</tr>
<tr>
<td>CaptureMicDevice</td>
<td>ICall</td>
</tr>
<tr>
<td>Conferenceld</td>
<td>ICall</td>
</tr>
<tr>
<td>DTMF</td>
<td>ICall</td>
</tr>
<tr>
<td>Duration</td>
<td>ICall</td>
</tr>
<tr>
<td>FailureReason</td>
<td>ICall</td>
</tr>
<tr>
<td>Finish()</td>
<td>ICall</td>
</tr>
<tr>
<td>Forward()</td>
<td>ICall</td>
</tr>
<tr>
<td>ForwardedBy</td>
<td>ICall</td>
</tr>
<tr>
<td>Hold()</td>
<td>ICall</td>
</tr>
<tr>
<td>Id</td>
<td>ICall</td>
</tr>
<tr>
<td>InputDevice</td>
<td>ICall</td>
</tr>
<tr>
<td>InputStatus</td>
<td>ICall</td>
</tr>
<tr>
<td>Join([in] LONG Id)</td>
<td>ICall</td>
</tr>
<tr>
<td>OutputDevice</td>
<td>ICall</td>
</tr>
<tr>
<td>Participants</td>
<td>ICall</td>
</tr>
<tr>
<td>PartnerDisplayName</td>
<td>ICall</td>
</tr>
<tr>
<td>PartnerHandle</td>
<td>ICall</td>
</tr>
<tr>
<td>PstnNumber</td>
<td>ICall</td>
</tr>
<tr>
<td>PstnStatus</td>
<td>ICall</td>
</tr>
<tr>
<td>Rate</td>
<td>ICall</td>
</tr>
<tr>
<td>RateCurrency</td>
<td>ICall</td>
</tr>
<tr>
<td>RatePrecision</td>
<td>ICall</td>
</tr>
<tr>
<td>RedirectToVoicemail()</td>
<td>ICall</td>
</tr>
<tr>
<td>Resume()</td>
<td>ICall</td>
</tr>
<tr>
<td>Seen</td>
<td>ICall</td>
</tr>
<tr>
<td>StartVideoReceive()</td>
<td>ICall</td>
</tr>
</tbody>
</table>


<table>
<thead>
<tr>
<th>Method/Property</th>
<th>Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>StartVideoSend()</td>
<td>ICall</td>
</tr>
<tr>
<td>Status</td>
<td>ICall</td>
</tr>
<tr>
<td>StopVideoReceive()</td>
<td>ICall</td>
</tr>
<tr>
<td>StopVideoSend()</td>
<td>ICall</td>
</tr>
<tr>
<td>Subject</td>
<td>ICall</td>
</tr>
<tr>
<td>TargetIdentity</td>
<td>ICall</td>
</tr>
<tr>
<td>Timestamp</td>
<td>ICall</td>
</tr>
<tr>
<td>Transfer([in] BSTR Target)</td>
<td>ICall</td>
</tr>
<tr>
<td>TransferActive</td>
<td>ICall</td>
</tr>
<tr>
<td>TransferredBy</td>
<td>ICall</td>
</tr>
<tr>
<td>TransferredTo</td>
<td>ICall</td>
</tr>
<tr>
<td>TransferStatus</td>
<td>ICall</td>
</tr>
<tr>
<td>Type</td>
<td>ICall</td>
</tr>
<tr>
<td>VideoReceiveStatus</td>
<td>ICall</td>
</tr>
<tr>
<td>VideoSendStatus</td>
<td>ICall</td>
</tr>
<tr>
<td>VideoStatus</td>
<td>ICall</td>
</tr>
<tr>
<td>VmAllowedDuration</td>
<td>ICall</td>
</tr>
<tr>
<td>VmDuration</td>
<td>ICall</td>
</tr>
</tbody>
</table>

Copyright Ⓡ 2006 Skype Limited. All rights reserved.
Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td>Class List</td>
<td>Class Members</td>
</tr>
</tbody>
</table>
# ICallChannel Member List

This is the complete list of members for **ICallChannel**, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Call</td>
<td>ICallChannel</td>
</tr>
<tr>
<td>Manager</td>
<td>ICallChannel</td>
</tr>
<tr>
<td>SendTextMessage(BSTR Text)</td>
<td>ICallChannel</td>
</tr>
<tr>
<td>Stream</td>
<td>ICallChannel</td>
</tr>
<tr>
<td>Type</td>
<td>ICallChannel</td>
</tr>
</tbody>
</table>

Copyright © 2006 Skype Limited. All rights reserved.
Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td>Class List</td>
<td>Class Members</td>
</tr>
</tbody>
</table>
ICallChannelCollection Member List

This is the complete list of members for **ICallChannelCollection**, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>_NewEnum</td>
<td>ICallChannelCollection</td>
</tr>
<tr>
<td>Add([in] ICallChannel *pChannel)</td>
<td>ICallChannelCollection</td>
</tr>
<tr>
<td>Count</td>
<td>ICallChannelCollection</td>
</tr>
<tr>
<td>Item</td>
<td>ICallChannelCollection</td>
</tr>
<tr>
<td>Remove([in] long Index)</td>
<td>ICallChannelCollection</td>
</tr>
<tr>
<td>RemoveAll(void)</td>
<td>ICallChannelCollection</td>
</tr>
</tbody>
</table>

Copyright © 2006 Skype Limited. All rights reserved.
# Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
# ICallChannelManager Member List

This is the complete list of members for [ICallChannelManager](#), including all inherited members.

<table>
<thead>
<tr>
<th>Member</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Channels</td>
<td>ICallChannelManager</td>
</tr>
<tr>
<td>ChannelType</td>
<td>ICallChannelManager</td>
</tr>
<tr>
<td>Connect([in] ISkype *pSkype)</td>
<td>ICallChannelManager</td>
</tr>
<tr>
<td>CreateApplication([in, defaultvalue(&quot;&quot;')] BSTR ApplicationName)</td>
<td>ICallChannelManager</td>
</tr>
<tr>
<td>Created</td>
<td>ICallChannelManager</td>
</tr>
<tr>
<td>Disconnect()</td>
<td>ICallChannelManager</td>
</tr>
<tr>
<td>Name</td>
<td>ICallChannelManager</td>
</tr>
</tbody>
</table>

Copyright ⚖ 2006 Skype Limited. All rights reserved.
Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td>Class List</td>
<td>Class Members</td>
</tr>
</tbody>
</table>

---

**Skype4COM 1.0.36.0**

- **Files**: Class List, Class Members
- **Examples**: Classes
- **Main Page**: Skype4COM 1.0.36.0
- **Related Pages**: Skype4COM 1.0.36.0
- **Modules**: Skype4COM 1.0.36.0
- **Classes**: Skype4COM 1.0.36.0
<table>
<thead>
<tr>
<th><strong>ICallChannelMessage Member List</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>This is the complete list of members for <strong>ICallChannelMessage</strong>, including all inherited members.</td>
</tr>
<tr>
<td><strong>Text</strong> <strong>ICallChannelMessage</strong></td>
</tr>
</tbody>
</table>

Copyright � 2006 Skype Limited. All rights reserved.
Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td>Class List</td>
<td>Class Members</td>
</tr>
</tbody>
</table>

**Files**

- Class List
- Class Members
ICallCollection Member List

This is the complete list of members for **ICallCollection**, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>_NewEnum</code></td>
<td>ICallCollection</td>
</tr>
<tr>
<td><code>Add([in] ICall *pCall)</code></td>
<td>ICallCollection</td>
</tr>
<tr>
<td><code>Count</code></td>
<td>ICallCollection</td>
</tr>
<tr>
<td><code>Item</code></td>
<td>ICallCollection</td>
</tr>
<tr>
<td><code>Remove([in] long Index)</code></td>
<td>ICallCollection</td>
</tr>
<tr>
<td><code>RemoveAll(void)</code></td>
<td>ICallCollection</td>
</tr>
</tbody>
</table>

Copyright © 2006 Skype Limited. All rights reserved.
<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td>Class List</td>
<td>Class Members</td>
</tr>
</tbody>
</table>
# IChat Member List

This is the complete list of members for **IChat**, including all inherited members.

<table>
<thead>
<tr>
<th>Method Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AcceptAdd(void)</td>
<td></td>
</tr>
<tr>
<td>ActiveMembers</td>
<td></td>
</tr>
<tr>
<td>ActivityTimestamp</td>
<td></td>
</tr>
<tr>
<td>Adder</td>
<td></td>
</tr>
<tr>
<td>AddMembers([in] IUserCollection *pMembers)</td>
<td></td>
</tr>
<tr>
<td>AlertString</td>
<td></td>
</tr>
<tr>
<td>Applicants</td>
<td></td>
</tr>
<tr>
<td>Blob</td>
<td></td>
</tr>
<tr>
<td>Bookmark()</td>
<td></td>
</tr>
<tr>
<td>Bookmarked</td>
<td></td>
</tr>
<tr>
<td>ClearRecentMessages(void)</td>
<td></td>
</tr>
<tr>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>DialogPartner</td>
<td></td>
</tr>
<tr>
<td>Disband(void)</td>
<td></td>
</tr>
<tr>
<td>EnterPassword([in] BSTR Password)</td>
<td></td>
</tr>
<tr>
<td>FriendlyName</td>
<td></td>
</tr>
<tr>
<td>GuideLines</td>
<td></td>
</tr>
<tr>
<td>Join(void)</td>
<td></td>
</tr>
<tr>
<td>Kick([in] BSTR Handle)</td>
<td></td>
</tr>
<tr>
<td>KickBan([in] BSTR Handle)</td>
<td></td>
</tr>
<tr>
<td>Leave()</td>
<td></td>
</tr>
<tr>
<td>MemberObjects</td>
<td></td>
</tr>
<tr>
<td>Members</td>
<td></td>
</tr>
<tr>
<td>Messages</td>
<td></td>
</tr>
<tr>
<td>MyRole</td>
<td></td>
</tr>
<tr>
<td>MyStatus</td>
<td></td>
</tr>
<tr>
<td>Name</td>
<td></td>
</tr>
<tr>
<td>OpenWindow()</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td><strong>Options</strong></td>
<td>IChat</td>
</tr>
<tr>
<td><strong>PasswordHint</strong></td>
<td>IChat</td>
</tr>
<tr>
<td><strong>Posters</strong></td>
<td>IChat</td>
</tr>
<tr>
<td><strong>RecentMessages</strong></td>
<td>IChat</td>
</tr>
<tr>
<td><strong>SendMessage</strong>([in] BSTR MessageText,[out, retval] IChatMessage **pVal))</td>
<td>IChat</td>
</tr>
<tr>
<td><strong>SetPassword</strong>([in] BSTR Password,[in, defaultvalue(&quot;&quot;)] BSTR Hint)</td>
<td>IChat</td>
</tr>
<tr>
<td><strong>Status</strong></td>
<td>IChat</td>
</tr>
<tr>
<td><strong>Timestamp</strong></td>
<td>IChat</td>
</tr>
<tr>
<td><strong>Topic</strong></td>
<td>IChat</td>
</tr>
<tr>
<td><strong>TopicXML</strong></td>
<td>IChat</td>
</tr>
<tr>
<td><strong>Type</strong></td>
<td>IChat</td>
</tr>
<tr>
<td><strong>Unbookmark()</strong></td>
<td>IChat</td>
</tr>
</tbody>
</table>

Copyright ⓒ 2006 Skype Limited. All rights reserved.
Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Files</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class List</td>
<td>Class Members</td>
</tr>
</tbody>
</table>
## IChatCollection Member List

This is the complete list of members for **IChatCollection**, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>_NewEnum</td>
<td>IChatCollection</td>
</tr>
<tr>
<td>Add([in] IChat *pItem)</td>
<td>IChatCollection</td>
</tr>
<tr>
<td>Count</td>
<td>IChatCollection</td>
</tr>
<tr>
<td>Item</td>
<td>IChatCollection</td>
</tr>
<tr>
<td>Remove([in] long Index)</td>
<td>IChatCollection</td>
</tr>
<tr>
<td>RemoveAll(void)</td>
<td>IChatCollection</td>
</tr>
</tbody>
</table>

Copyright © 2006 Skype Limited. All rights reserved.
Skype4COM 1.0.36.0

Main Page | Related Pages | Modules | Classes
---|---|---|---
Files | Examples
---|---
Class List | Class Members
# IChatMember Member List

This is the complete list of members for `IChatMember`, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>CanSetRoleTo</td>
<td><code>IChatMember</code></td>
</tr>
<tr>
<td>Chat</td>
<td><code>IChatMember</code></td>
</tr>
<tr>
<td>Handle</td>
<td><code>IChatMember</code></td>
</tr>
<tr>
<td>Id</td>
<td><code>IChatMember</code></td>
</tr>
<tr>
<td>IsActive</td>
<td><code>IChatMember</code></td>
</tr>
<tr>
<td>Role</td>
<td><code>IChatMember</code></td>
</tr>
</tbody>
</table>

Copyright © 2006 Skype Limited. All rights reserved.
## Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td>Class List</td>
<td>Class Members</td>
</tr>
</tbody>
</table>

**Skype4COM 1.0.36.0**
IChatMemberCollection Member List

This is the complete list of members for **IChatMemberCollection**, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>_NewEnum</td>
<td>IChatMemberCollection</td>
</tr>
<tr>
<td>Count</td>
<td>IChatMemberCollection</td>
</tr>
<tr>
<td>Item</td>
<td>IChatMemberCollection</td>
</tr>
</tbody>
</table>

Copyright © 2006 Skype Limited. All rights reserved.
Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
# IChatMessage Member List

This is the complete list of members for `IChatMessage`, including all inherited members.

<table>
<thead>
<tr>
<th>Member</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Body</td>
<td>IChatMessage</td>
</tr>
<tr>
<td>Chat</td>
<td>IChatMessage</td>
</tr>
<tr>
<td>ChatName</td>
<td>IChatMessage</td>
</tr>
<tr>
<td>EditedBy</td>
<td>IChatMessage</td>
</tr>
<tr>
<td>EditedTimestamp</td>
<td>IChatMessage</td>
</tr>
<tr>
<td>FromDisplayName</td>
<td>IChatMessage</td>
</tr>
<tr>
<td>FromHandle</td>
<td>IChatMessage</td>
</tr>
<tr>
<td>Id</td>
<td>IChatMessage</td>
</tr>
<tr>
<td>IsEditable</td>
<td>IChatMessage</td>
</tr>
<tr>
<td>LeaveReason</td>
<td>IChatMessage</td>
</tr>
<tr>
<td>Options</td>
<td>IChatMessage</td>
</tr>
<tr>
<td>Role</td>
<td>IChatMessage</td>
</tr>
<tr>
<td>Seen</td>
<td>IChatMessage</td>
</tr>
<tr>
<td>Sender</td>
<td>IChatMessage</td>
</tr>
<tr>
<td>Status</td>
<td>IChatMessage</td>
</tr>
<tr>
<td>Timestamp</td>
<td>IChatMessage</td>
</tr>
<tr>
<td>Type</td>
<td>IChatMessage</td>
</tr>
<tr>
<td>Users</td>
<td>IChatMessage</td>
</tr>
</tbody>
</table>
Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td>Class List</td>
<td>Class Members</td>
</tr>
</tbody>
</table>
IChatMessageCollection Member List

This is the complete list of members for IChatMessageCollection, including all inherited members.

<table>
<thead>
<tr>
<th>Member</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>_NewEnum</td>
<td>IChatMessageCollection</td>
</tr>
<tr>
<td>Add([in] IChatMessage *pItem)</td>
<td>IChatMessageCollection</td>
</tr>
<tr>
<td>Count</td>
<td>IChatMessageCollection</td>
</tr>
<tr>
<td>Item</td>
<td>IChatMessageCollection</td>
</tr>
<tr>
<td>Remove([in] long Index)</td>
<td>IChatMessageCollection</td>
</tr>
<tr>
<td>RemoveAll(void)</td>
<td>IChatMessageCollection</td>
</tr>
</tbody>
</table>

Copyright © 2006 Skype Limited. All rights reserved.
<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
# IClient Member List

This is the complete list of members for **IClient**, including all inherited members.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ButtonPressed</strong> ([in] BSTR Key)</td>
<td>IClient</td>
</tr>
<tr>
<td><strong>ButtonReleased</strong> ([in] BSTR Key)</td>
<td>IClient</td>
</tr>
<tr>
<td><strong>CreateEvent</strong> ([in] BSTR EventId,[in] BSTR Caption,[in] BSTR Hint,[out, ret] IPluginEvent **pEvent)</td>
<td>IClient</td>
</tr>
<tr>
<td><strong>CreateMenuItem</strong> ([in] BSTR MenuItemId,[in] TPluginContext PluginContext,[in] BSTR CaptionText,[in, defaultvalue(&quot;&quot;)] BSTR HintText,[in, defaultvalue(&quot;&quot;)] BSTR IconPath,[in, defaultvalue(-1)] VARIANT_BOOL Enabled,[in, defaultvalue(pluginContactTypeAll)] TPluginContactType ContactType,[in, defaultvalue(0)] VARIANT_BOOL MultipleContacts,[out, ret] IPluginMenuItem **pMenuItem)</td>
<td>IClient</td>
</tr>
<tr>
<td><strong>Focus</strong> ()</td>
<td>IClient</td>
</tr>
<tr>
<td><strong>IsRunning</strong></td>
<td>IClient</td>
</tr>
<tr>
<td><strong>Minimize</strong> ()</td>
<td>IClient</td>
</tr>
<tr>
<td><strong>OpenAddContactDialog</strong> ([in, defaultvalue(&quot;&quot;)] BSTR Username)</td>
<td>IClient</td>
</tr>
<tr>
<td><strong>OpenAuthorizationDialog</strong> ([in] BSTR Username)</td>
<td>IClient</td>
</tr>
<tr>
<td><strong>OpenBlockedUsersDialog</strong> ()</td>
<td>IClient</td>
</tr>
<tr>
<td><strong>OpenCallHistoryTab</strong> ()</td>
<td>IClient</td>
</tr>
<tr>
<td><strong>OpenConferenceDialog</strong> ()</td>
<td>IClient</td>
</tr>
<tr>
<td><strong>OpenContactsTab</strong> ()</td>
<td>IClient</td>
</tr>
<tr>
<td><strong>OpenDialog</strong> ([in] BSTR Name,[in, defaultvalue(&quot;&quot;)] BSTR Param1,[in, defaultvalue(&quot;&quot;)] BSTR Param2)</td>
<td>IClient</td>
</tr>
<tr>
<td><strong>OpenDialpadTab</strong> ()</td>
<td>IClient</td>
</tr>
<tr>
<td><strong>OpenFileTransferDialog</strong> ([in] BSTR User,[in] BSTR Folder)</td>
<td>IClient</td>
</tr>
<tr>
<td><strong>OpenGettingStartedWizard</strong> ()</td>
<td>IClient</td>
</tr>
<tr>
<td><strong>OpenImportContactsWizard</strong> ()</td>
<td>IClient</td>
</tr>
<tr>
<td><strong>OpenLiveTab</strong> ()</td>
<td>IClient</td>
</tr>
<tr>
<td><strong>OpenMessageDialog</strong> ([in] BSTR Username,[in,</td>
<td>IClient</td>
</tr>
<tr>
<td>Method</td>
<td>Return Type</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>defaultvalue(&quot;&quot;) BSTR Text</td>
<td>IClient</td>
</tr>
<tr>
<td>OpenOptionsDialog([in] BSTR Page)</td>
<td>IClient</td>
</tr>
<tr>
<td>OpenProfileDialog()</td>
<td>IClient</td>
</tr>
<tr>
<td>OpenSearchDialog()</td>
<td>IClient</td>
</tr>
<tr>
<td>OpenSendContactsDialog([in, defaultvalue(&quot;&quot;) BSTR Username)</td>
<td>IClient</td>
</tr>
<tr>
<td>OpenSmsDialog([in] BSTR SmsId)</td>
<td>IClient</td>
</tr>
<tr>
<td>OpenUserInfoDialog([in] BSTR Username)</td>
<td>IClient</td>
</tr>
<tr>
<td>OpenVideoTestDialog()</td>
<td>IClient</td>
</tr>
<tr>
<td>Shutdown()</td>
<td>IClient</td>
</tr>
<tr>
<td>Start([in, defaultvalue(0)] VARIANT_BOOL Minimized,[in, defaultvalue(0)] VARIANT_BOOL Nosplash)</td>
<td>IClient</td>
</tr>
<tr>
<td>Wallpaper</td>
<td>IClient</td>
</tr>
</tbody>
</table>

Copyright ⓙ 2006 Skype Limited. All rights reserved.
Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Files</strong></td>
<td><strong>Examples</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
# ICommand Member List

This is the complete list of members for **ICommand**, including all inherited members.

<table>
<thead>
<tr>
<th>Member</th>
<th>ICommand</th>
</tr>
</thead>
<tbody>
<tr>
<td>Blocking</td>
<td>ICommand</td>
</tr>
<tr>
<td>Command</td>
<td>ICommand</td>
</tr>
<tr>
<td>Expected</td>
<td>ICommand</td>
</tr>
<tr>
<td>Id</td>
<td>ICommand</td>
</tr>
<tr>
<td>Reply</td>
<td>ICommand</td>
</tr>
<tr>
<td>Timeout</td>
<td>ICommand</td>
</tr>
</tbody>
</table>

Copyright © 2006 Skype Limited. All rights reserved.
Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
IConference Member List

This is the complete list of members for IConference, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ActiveCalls</td>
<td>IConference</td>
</tr>
<tr>
<td>Calls</td>
<td>IConference</td>
</tr>
<tr>
<td>Finish()</td>
<td>IConference</td>
</tr>
<tr>
<td>Hold()</td>
<td>IConference</td>
</tr>
<tr>
<td>Id</td>
<td>IConference</td>
</tr>
<tr>
<td>Resume()</td>
<td>IConference</td>
</tr>
</tbody>
</table>

Copyright © 2006 Skype Limited. All rights reserved.
Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
This is the complete list of members for `IConferenceCollection`, including all inherited members.

<table>
<thead>
<tr>
<th>Member</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>_NewEnum</code></td>
<td><code>IConferenceCollection</code></td>
</tr>
<tr>
<td><code>Add([in] IConference * pItem)</code></td>
<td><code>IConferenceCollection</code></td>
</tr>
<tr>
<td><code>Count</code></td>
<td><code>IConferenceCollection</code></td>
</tr>
<tr>
<td><code>Item</code></td>
<td><code>IConferenceCollection</code></td>
</tr>
<tr>
<td><code>Remove([in] long Index)</code></td>
<td><code>IConferenceCollection</code></td>
</tr>
<tr>
<td><code>RemoveAll(void)</code></td>
<td><code>IConferenceCollection</code></td>
</tr>
</tbody>
</table>

Copyright © 2006 Skype Limited. All rights reserved.
Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
# IConversion Member List

This is the complete list of members for **IConversion**, including all inherited members.

<table>
<thead>
<tr>
<th>Method Name</th>
<th>Parameters</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>AttachmentStatusToText</td>
<td>([in] TAttachmentStatus status,[out, retval] BSTR *pVal)</td>
<td>IConversion</td>
</tr>
<tr>
<td>BuddyStatusToText</td>
<td>([in] TBuddyStatus status,[out, retval] BSTR *pVal)</td>
<td>IConversion</td>
</tr>
<tr>
<td>CallFailureReasonToText</td>
<td>([in] TCallFailureReason reason,[out, retval] BSTR *pVal)</td>
<td>IConversion</td>
</tr>
<tr>
<td>CallStatusToText</td>
<td>([in] TCallStatus Status,[out, retval] BSTR *pVal)</td>
<td>IConversion</td>
</tr>
<tr>
<td>CallTypeToText</td>
<td>([in] TCallType CallType,[out, retval] BSTR *pVal)</td>
<td>IConversion</td>
</tr>
<tr>
<td>CallVideoSendStatusToText</td>
<td>([in] TCallVideoSendStatus Status,[out, retval] BSTR *pVal)</td>
<td>IConversion</td>
</tr>
<tr>
<td>CallVideoStatusToText</td>
<td>([in] TCallVideoStatus Status,[out, retval] BSTR *pVal)</td>
<td>IConversion</td>
</tr>
<tr>
<td>ChatLeaveReasonToText</td>
<td>([in] TChatLeaveReason reason,[out, retval] BSTR *pVal)</td>
<td>IConversion</td>
</tr>
<tr>
<td>ChatMessageStatusToText</td>
<td>([in] TChatMessageStatus Status,[out, retval] BSTR *pVal)</td>
<td>IConversion</td>
</tr>
<tr>
<td>ChatMessageTypeToText</td>
<td>([in] TChatMessageType Type,[out, retval] BSTR *pVal)</td>
<td>IConversion</td>
</tr>
<tr>
<td>ChatStatusToText</td>
<td>([in] TChatStatus status,[out, retval] BSTR *pVal)</td>
<td>IConversion</td>
</tr>
<tr>
<td>ConnectionStatusToText</td>
<td>([in] TConnectionStatus Status,[out, retval] BSTR *pVal)</td>
<td>IConversion</td>
</tr>
<tr>
<td>GroupTypeToText</td>
<td>([in] TGroupType Type,[out, retval] BSTR *pVal)</td>
<td>IConversion</td>
</tr>
<tr>
<td>Language</td>
<td></td>
<td>IConversion</td>
</tr>
<tr>
<td>OnlineStatusToText</td>
<td>([in] TOnlineStatus status,[out, retval] BSTR *pVal)</td>
<td>IConversion</td>
</tr>
<tr>
<td>SmsMessageStatusToText</td>
<td>([in] TSmsMessageStatus Status,[out, retval] BSTR *pVal)</td>
<td>IConversion</td>
</tr>
<tr>
<td>Function</td>
<td>Parameters</td>
<td>Remarks</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>SmsMessageTypeToText</td>
<td>([in] TSmsMessageType type, [out, retval] BSTR *pVal)</td>
<td>IConversion</td>
</tr>
<tr>
<td>SmsTargetStatusToText</td>
<td>([in] TSmsTargetStatus status, [out, retval] BSTR *pVal)</td>
<td>IConversion</td>
</tr>
<tr>
<td>TextToAttachmentStatus</td>
<td>([in] BSTR Text, [out, retval] TAttachmentStatus *pVal)</td>
<td>IConversion</td>
</tr>
<tr>
<td>TextToBuddyStatus</td>
<td>([in] BSTR Text, [out, retval] TBuddyStatus *pVal)</td>
<td>IConversion</td>
</tr>
<tr>
<td>TextToCallStatus</td>
<td>([in] BSTR Text, [out, retval] TCallStatus *pVal)</td>
<td>IConversion</td>
</tr>
<tr>
<td>TextToCallType</td>
<td>([in] BSTR Text, [out, retval] TCallType *pVal)</td>
<td>IConversion</td>
</tr>
<tr>
<td>TextToChatMessageStatus</td>
<td>([in] BSTR Text, [out, retval] TChatMessageStatus *pVal)</td>
<td>IConversion</td>
</tr>
<tr>
<td>TextToChatMessageType</td>
<td>([in] BSTR Text, [out, retval] TChatMessageType *pVal)</td>
<td>IConversion</td>
</tr>
<tr>
<td>TextToConnectionStatus</td>
<td>([in] BSTR Text, [out, retval] TConnectionStatus *pVal)</td>
<td>IConversion</td>
</tr>
<tr>
<td>TextToGroupType</td>
<td>([in] BSTR Text, [out, retval] TGroupType *pVal)</td>
<td>IConversion</td>
</tr>
<tr>
<td>TextToOnlineStatus</td>
<td>([in] BSTR Text, [out, retval] TOnlineStatus *pVal)</td>
<td>IConversion</td>
</tr>
<tr>
<td>TextToUserSex</td>
<td>([in] BSTR Text, [out, retval] TUserSex *pVal)</td>
<td>IConversion</td>
</tr>
<tr>
<td>TextToUserStatus</td>
<td>([in] BSTR Text, [out, retval] TUserStatus *pVal)</td>
<td>IConversion</td>
</tr>
<tr>
<td>TextToVoicemailStatus</td>
<td>([in] BSTR Text, [out, retval] TVoicemailStatus *pVal)</td>
<td>IConversion</td>
</tr>
<tr>
<td>UserSexToText</td>
<td>([in] TUserSex Sex, [out, retval] BSTR *pVal)</td>
<td>IConversion</td>
</tr>
<tr>
<td>UserStatusToText</td>
<td>([in] TUserStatus Status, [out, retval] BSTR *pVal)</td>
<td>IConversion</td>
</tr>
<tr>
<td>VoicemailFailureReasonToText</td>
<td>([in] TVoicemailFailureReason code, [out, retval] BSTR *pVal)</td>
<td>IConversion</td>
</tr>
<tr>
<td>VoicemailStatusToText</td>
<td>([in] TVoicemailStatus status,</td>
<td>IConversion</td>
</tr>
</tbody>
</table>
[out, retval] BSTR *pVal)

VoicemailTypeToText([in] TVoicemailType type,[out, retval] BSTR *pVal)
### IFileTransfer Member List

This is the complete list of members for **IFileTransfer**, including all inherited members.

<table>
<thead>
<tr>
<th>Member</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>BytesPerSecond</td>
<td>IFileTransfer</td>
</tr>
<tr>
<td>BytesTransferred</td>
<td>IFileTransfer</td>
</tr>
<tr>
<td>FailureReason</td>
<td>IFileTransfer</td>
</tr>
<tr>
<td>FileName</td>
<td>IFileTransfer</td>
</tr>
<tr>
<td>FilePath</td>
<td>IFileTransfer</td>
</tr>
<tr>
<td>FinishTime</td>
<td>IFileTransfer</td>
</tr>
<tr>
<td>Id</td>
<td>IFileTransfer</td>
</tr>
<tr>
<td>PartnerDisplayName</td>
<td>IFileTransfer</td>
</tr>
<tr>
<td>PartnerHandle</td>
<td>IFileTransfer</td>
</tr>
<tr>
<td>StartTime</td>
<td>IFileTransfer</td>
</tr>
<tr>
<td>Status</td>
<td>IFileTransfer</td>
</tr>
<tr>
<td>Type</td>
<td>IFileTransfer</td>
</tr>
</tbody>
</table>

Copyright © 2006 Skype Limited. All rights reserved.
<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
**IFileTransferCollection Member List**

This is the complete list of members for **IFileTransferCollection**, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>_NewEnum</td>
<td>IFileTransferCollection</td>
</tr>
<tr>
<td>Add</td>
<td>IFileTransferCollection</td>
</tr>
<tr>
<td>Count</td>
<td>IFileTransferCollection</td>
</tr>
<tr>
<td>Item</td>
<td>IFileTransferCollection</td>
</tr>
<tr>
<td>Remove</td>
<td>IFileTransferCollection</td>
</tr>
<tr>
<td>RemoveAll</td>
<td>IFileTransferCollection</td>
</tr>
</tbody>
</table>

Copyright © 2006 Skype Limited. All rights reserved.
## IGroup Member List

This is the complete list of members for **IGroup**, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Accept()</strong></td>
<td>IGroup</td>
</tr>
<tr>
<td><strong>AddUser([in] BSTR Username)</strong></td>
<td>IGroup</td>
</tr>
<tr>
<td><strong>CustomGroupId</strong></td>
<td>IGroup</td>
</tr>
<tr>
<td><strong>Decline()</strong></td>
<td>IGroup</td>
</tr>
<tr>
<td><strong>DisplayName</strong></td>
<td>IGroup</td>
</tr>
<tr>
<td><strong>Id</strong></td>
<td>IGroup</td>
</tr>
<tr>
<td><strong>IsExpanded</strong></td>
<td>IGroup</td>
</tr>
<tr>
<td><strong>IsVisible</strong></td>
<td>IGroup</td>
</tr>
<tr>
<td><strong>OnlineUsers</strong></td>
<td>IGroup</td>
</tr>
<tr>
<td><strong>RemoveUser([in] BSTR Username)</strong></td>
<td>IGroup</td>
</tr>
<tr>
<td><strong>Share([in, defaultvalue(&quot;&quot;')] BSTR MessageText)</strong></td>
<td>IGroup</td>
</tr>
<tr>
<td><strong>Type</strong></td>
<td>IGroup</td>
</tr>
<tr>
<td><strong>Users</strong></td>
<td>IGroup</td>
</tr>
</tbody>
</table>

Copyright © 2006 Skype Limited. All rights reserved.
Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## IGroupCollection Member List

This is the complete list of members for **IGroupCollection**, including all inherited members.

<table>
<thead>
<tr>
<th>Member</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>_NewEnum</td>
<td>IGroupCollection</td>
</tr>
<tr>
<td>Add([in] IGroup *pGroup)</td>
<td>IGroupCollection</td>
</tr>
<tr>
<td>Count</td>
<td>IGroupCollection</td>
</tr>
<tr>
<td>Item</td>
<td>IGroupCollection</td>
</tr>
<tr>
<td>Remove([in] long Index)</td>
<td>IGroupCollection</td>
</tr>
<tr>
<td>RemoveAll(void)</td>
<td>IGroupCollection</td>
</tr>
</tbody>
</table>

Copyright © 2006 Skype Limited. All rights reserved.
Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Files</strong></td>
<td><strong>Examples</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
# IParticipant Member List

This is the complete list of members for IParticipant, including all inherited members.

<table>
<thead>
<tr>
<th>CallStatus</th>
<th>IParticipant</th>
</tr>
</thead>
<tbody>
<tr>
<td>CallType</td>
<td>IParticipant</td>
</tr>
<tr>
<td>DisplayName</td>
<td>IParticipant</td>
</tr>
<tr>
<td>Handle</td>
<td>IParticipant</td>
</tr>
</tbody>
</table>

Copyright ✨ 2006 Skype Limited. All rights reserved.
## Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### IParticipantCollection Member List

This is the complete list of members for **IParticipantCollection**, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>_NewEnum</td>
<td>IParticipantCollection</td>
</tr>
<tr>
<td>Add([in] IParticipant *pParticipant)</td>
<td>IParticipantCollection</td>
</tr>
<tr>
<td>Count</td>
<td>IParticipantCollection</td>
</tr>
<tr>
<td>Item</td>
<td>IParticipantCollection</td>
</tr>
<tr>
<td>Remove([in] long Index)</td>
<td>IParticipantCollection</td>
</tr>
<tr>
<td>RemoveAll(void)</td>
<td>IParticipantCollection</td>
</tr>
</tbody>
</table>

Copyright © 2006 Skype Limited. All rights reserved.
Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
IPluginEvent Member List

This is the complete list of members for IPluginEvent, including all inherited members.

<table>
<thead>
<tr>
<th>Member</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Delete</td>
<td>(void) IPluginEvent</td>
</tr>
<tr>
<td>Id</td>
<td>IPluginEvent</td>
</tr>
</tbody>
</table>

Copyright © 2006 Skype Limited. All rights reserved.
# Skype4COM 1.0.36.0

## Classes

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## IPluginMenuItem Member List

This is the complete list of members for **IPluginMenuItem**, including all inherited members.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Caption</td>
<td>IPluginMenuItem</td>
</tr>
<tr>
<td>Delete</td>
<td>(void) IPluginMenuItem</td>
</tr>
<tr>
<td>Enabled</td>
<td>IPluginMenuItem</td>
</tr>
<tr>
<td>Hint</td>
<td>IPluginMenuItem</td>
</tr>
<tr>
<td>Id</td>
<td>IPluginMenuItem</td>
</tr>
</tbody>
</table>

Copyright © 2006 Skype Limited. All rights reserved.
<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td>Class List</td>
<td>Class Members</td>
</tr>
</tbody>
</table>
**IProfile Member List**

This is the complete list of members for **IProfile**, including all inherited members.

<table>
<thead>
<tr>
<th>Member</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>About</td>
<td>IProfile</td>
</tr>
<tr>
<td>Balance</td>
<td>IProfile</td>
</tr>
<tr>
<td>BalanceCurrency</td>
<td>IProfile</td>
</tr>
<tr>
<td>BalanceToText</td>
<td>IProfile</td>
</tr>
<tr>
<td>Birthday</td>
<td>IProfile</td>
</tr>
<tr>
<td>CallApplyCF</td>
<td>IProfile</td>
</tr>
<tr>
<td>CallForwardRules</td>
<td>IProfile</td>
</tr>
<tr>
<td>CallNoAnswerTimeout</td>
<td>IProfile</td>
</tr>
<tr>
<td>CallSendToVM</td>
<td>IProfile</td>
</tr>
<tr>
<td>City</td>
<td>IProfile</td>
</tr>
<tr>
<td>Country</td>
<td>IProfile</td>
</tr>
<tr>
<td>FullName</td>
<td>IProfile</td>
</tr>
<tr>
<td>Homepage</td>
<td>IProfile</td>
</tr>
<tr>
<td>IPCountry</td>
<td>IProfile</td>
</tr>
<tr>
<td>Languages</td>
<td>IProfile</td>
</tr>
<tr>
<td>MoodText</td>
<td>IProfile</td>
</tr>
<tr>
<td>PhoneHome</td>
<td>IProfile</td>
</tr>
<tr>
<td>PhoneMobile</td>
<td>IProfile</td>
</tr>
<tr>
<td>PhoneOffice</td>
<td>IProfile</td>
</tr>
<tr>
<td>Province</td>
<td>IProfile</td>
</tr>
<tr>
<td>RichMoodText</td>
<td>IProfile</td>
</tr>
<tr>
<td>Sex</td>
<td>IProfile</td>
</tr>
<tr>
<td>Timezone</td>
<td>IProfile</td>
</tr>
<tr>
<td>ValidatedSmsNumbers</td>
<td>IProfile</td>
</tr>
</tbody>
</table>

Copyright © 2006 Skype Limited. All rights reserved.
<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Skype4COM 1.0.36.0**
# ISettings Member List

This is the complete list of members for **ISettings**, including all inherited members.

<table>
<thead>
<tr>
<th>Member</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>AEC</td>
<td>ISettings</td>
</tr>
<tr>
<td>AGC</td>
<td>ISettings</td>
</tr>
<tr>
<td>AudioIn</td>
<td>ISettings</td>
</tr>
<tr>
<td>AudioOut</td>
<td>ISettings</td>
</tr>
<tr>
<td>AutoAway</td>
<td>ISettings</td>
</tr>
<tr>
<td>Avatar</td>
<td>ISettings</td>
</tr>
<tr>
<td>Language</td>
<td>ISettings</td>
</tr>
<tr>
<td><strong>LoadAvatarFromFile</strong></td>
<td>ISettings</td>
</tr>
<tr>
<td></td>
<td>([in] BSTR Filename,[in, defaultvalue(&quot;1&quot;)</td>
</tr>
<tr>
<td>PCSpeaker</td>
<td>ISettings</td>
</tr>
<tr>
<td>ResetIdleTimer</td>
<td>ISettings</td>
</tr>
<tr>
<td>Ringer</td>
<td>ISettings</td>
</tr>
<tr>
<td>RingTone</td>
<td>ISettings</td>
</tr>
<tr>
<td>RingToneStatus</td>
<td>ISettings</td>
</tr>
<tr>
<td><strong>SaveAvatarToFile</strong></td>
<td>ISettings</td>
</tr>
<tr>
<td></td>
<td>([in] BSTR Filename,[in, defaultvalue(&quot;1&quot;)</td>
</tr>
<tr>
<td>VideoIn</td>
<td>ISettings</td>
</tr>
</tbody>
</table>

---

Copyright � 2006 Skype Limited. All rights reserved.
Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td>Classes</td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
# ISkype Member List

This is the complete list of members for **ISkype**, including all inherited members.

<table>
<thead>
<tr>
<th>Member</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>ActiveCalls</td>
<td>ISkype</td>
</tr>
<tr>
<td>ActiveChats</td>
<td>ISkype</td>
</tr>
<tr>
<td>ActiveFileTransfers</td>
<td>ISkype</td>
</tr>
<tr>
<td>ApiSecurityContextEnabled</td>
<td>ISkype</td>
</tr>
<tr>
<td>ApiWrapperVersion</td>
<td>ISkype</td>
</tr>
<tr>
<td>Application</td>
<td>ISkype</td>
</tr>
<tr>
<td>AsyncSearchUsers</td>
<td>ISkype</td>
</tr>
<tr>
<td>Attach</td>
<td>ISkype</td>
</tr>
<tr>
<td>AttachmentStatus</td>
<td>ISkype</td>
</tr>
<tr>
<td>BookmarkedChats</td>
<td>ISkype</td>
</tr>
<tr>
<td>Cache</td>
<td>ISkype</td>
</tr>
<tr>
<td>Call</td>
<td>ISkype</td>
</tr>
<tr>
<td>Calls</td>
<td>ISkype</td>
</tr>
<tr>
<td>ChangeUserStatus</td>
<td>ISkype</td>
</tr>
<tr>
<td>Chat</td>
<td>ISkype</td>
</tr>
<tr>
<td>Chats</td>
<td>ISkype</td>
</tr>
<tr>
<td>ClearCallHistory</td>
<td>ISkype</td>
</tr>
<tr>
<td>ClearChatHistory</td>
<td>ISkype</td>
</tr>
<tr>
<td>ClearVoicemailHistory</td>
<td>ISkype</td>
</tr>
<tr>
<td>Client</td>
<td>ISkype</td>
</tr>
<tr>
<td>Command</td>
<td>ISkype</td>
</tr>
<tr>
<td>CommandId</td>
<td>ISkype</td>
</tr>
<tr>
<td>Conference</td>
<td>ISkype</td>
</tr>
<tr>
<td>Conferences</td>
<td>ISkype</td>
</tr>
<tr>
<td>ConnectionStatus</td>
<td>ISkype</td>
</tr>
<tr>
<td>Method</td>
<td>Interface</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>Convert</td>
<td>ISkype</td>
</tr>
<tr>
<td><strong>CreateChatMultiple</strong>([in] IUserCollection *pMembers,[out, retval] IChat **pChat)</td>
<td>ISkype</td>
</tr>
<tr>
<td><strong>CreateChatUsingBlob</strong>([in] BSTR Blob,[out, retval] IChat **pChat)</td>
<td>ISkype</td>
</tr>
<tr>
<td><strong>CreateChatWith</strong>([in] BSTR Username,[out, retval] IChat **pVal)</td>
<td>ISkype</td>
</tr>
<tr>
<td><strong>CreateGroup</strong>([in] BSTR GroupName,[out, retval] IGroup **pGroup)</td>
<td>ISkype</td>
</tr>
<tr>
<td><strong>CreateSms</strong>([in] TSmsMessageType MessageType,[in] BSTR TargetNumbers,[out, retval] ISmsMessage **pMessage)</td>
<td>ISkype</td>
</tr>
<tr>
<td><strong>CurrentUser</strong></td>
<td>ISkype</td>
</tr>
<tr>
<td><strong>CurrentUserHandle</strong></td>
<td>ISkype</td>
</tr>
<tr>
<td><strong>CurrentUserProfile</strong></td>
<td>ISkype</td>
</tr>
<tr>
<td><strong>CurrentUserStatus</strong></td>
<td>ISkype</td>
</tr>
<tr>
<td><strong>CustomGroups</strong></td>
<td>ISkype</td>
</tr>
<tr>
<td><strong>DeleteGroup</strong>([in] LONG GroupId)</td>
<td>ISkype</td>
</tr>
<tr>
<td><strong>EnableApiSecurityContext</strong>([in] TApiSecurityContext Context)</td>
<td>ISkype</td>
</tr>
<tr>
<td><strong>FileTransfers</strong></td>
<td>ISkype</td>
</tr>
<tr>
<td><strong>FindChatUsingBlob</strong>([in] BSTR Blob,[out, retval] IChat **pChat)</td>
<td>ISkype</td>
</tr>
<tr>
<td><strong>FocusedContacts</strong></td>
<td>ISkype</td>
</tr>
<tr>
<td><strong>FriendlyName</strong></td>
<td>ISkype</td>
</tr>
<tr>
<td><strong>Friends</strong></td>
<td>ISkype</td>
</tr>
<tr>
<td><strong>Greeting</strong></td>
<td>ISkype</td>
</tr>
<tr>
<td><strong>Groups</strong></td>
<td>ISkype</td>
</tr>
<tr>
<td><strong>HardwiredGroups</strong></td>
<td>ISkype</td>
</tr>
<tr>
<td><strong>Message</strong></td>
<td>ISkype</td>
</tr>
<tr>
<td><strong>Messages</strong></td>
<td>ISkype</td>
</tr>
<tr>
<td><strong>MissedCalls</strong></td>
<td>ISkype</td>
</tr>
<tr>
<td><strong>MissedChats</strong></td>
<td>ISkype</td>
</tr>
<tr>
<td><strong>MissedMessages</strong></td>
<td>ISkype</td>
</tr>
<tr>
<td><strong>MissedSmss</strong></td>
<td>ISkype</td>
</tr>
<tr>
<td>MissedVoicemails</td>
<td>ISkype</td>
</tr>
<tr>
<td>Mute</td>
<td>ISkype</td>
</tr>
<tr>
<td>PlaceCall([in] BSTR Target,[in, defaultvalue('&quot;')] BSTR Target2,[in, defaultvalue('&quot;')] BSTR Target3,[in, defaultvalue('&quot;')] BSTR Target4,[out, retval] ICall **pCall)</td>
<td>ISkype</td>
</tr>
<tr>
<td>PredictiveDialerCountry</td>
<td>ISkype</td>
</tr>
<tr>
<td>Privilege</td>
<td>ISkype</td>
</tr>
<tr>
<td>Profile</td>
<td>ISkype</td>
</tr>
<tr>
<td>Property</td>
<td>ISkype</td>
</tr>
<tr>
<td>Protocol</td>
<td>ISkype</td>
</tr>
<tr>
<td>RecentChats</td>
<td>ISkype</td>
</tr>
<tr>
<td>ResetCache()</td>
<td>ISkype</td>
</tr>
<tr>
<td>SearchForUsers([in] BSTR Target,[out, retval] IUserCollection **pVal)</td>
<td>ISkype</td>
</tr>
<tr>
<td>SendCommand([in] ICommand *pCommand)</td>
<td>ISkype</td>
</tr>
<tr>
<td>SendMessage([in] BSTR Username,[in] BSTR Text,[out, retval] IChatMessage **pMessage)</td>
<td>ISkype</td>
</tr>
<tr>
<td>SendSms([in] BSTR TargetNumbers,[in] BSTR MessageText, [in, defaultvalue('&quot;')] BSTR ReplyToNumber,[out, retval] ISmsMessage **pMessage)</td>
<td>ISkype</td>
</tr>
<tr>
<td>SendVoicemail([in] BSTR Username,[out, retval] IVoicemail **pVoicemail)</td>
<td>ISkype</td>
</tr>
<tr>
<td>Settings</td>
<td>ISkype</td>
</tr>
<tr>
<td>SilentMode</td>
<td>ISkype</td>
</tr>
<tr>
<td>Smss</td>
<td>ISkype</td>
</tr>
<tr>
<td>Timeout</td>
<td>ISkype</td>
</tr>
<tr>
<td>User</td>
<td>ISkype</td>
</tr>
<tr>
<td>UsersWaitingAuthorization</td>
<td>ISkype</td>
</tr>
<tr>
<td>Variable</td>
<td>ISkype</td>
</tr>
<tr>
<td>Version</td>
<td>ISkype</td>
</tr>
<tr>
<td>Voicemail</td>
<td>ISkype</td>
</tr>
<tr>
<td>Voicemails</td>
<td>ISkype</td>
</tr>
</tbody>
</table>

Copyright � 2006 Skype Limited. All rights reserved.
# Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Files</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class List</td>
<td>Class Members</td>
</tr>
</tbody>
</table>

## Classes

### Main Page

### Related Pages

### Modules

### Classes
## ISmsChunk Member List

This is the complete list of members for ISmsChunk, including all inherited members.

<table>
<thead>
<tr>
<th>Member</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>CharactersLeft</td>
<td>ISmsChunk</td>
</tr>
<tr>
<td>Id</td>
<td>ISmsChunk</td>
</tr>
<tr>
<td>Message</td>
<td>ISmsChunk</td>
</tr>
<tr>
<td>Text</td>
<td>ISmsChunk</td>
</tr>
</tbody>
</table>

Copyright © 2006 Skype Limited. All rights reserved.
This is the complete list of members for `ISmsChunkCollection`, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>_NewEnum</code></td>
<td><code>ISmsChunkCollection</code></td>
</tr>
<tr>
<td><code>Add</code></td>
<td><code>ISmsChunkCollection</code> ([in] ISmsChunk *pItem)</td>
</tr>
<tr>
<td><code>Count</code></td>
<td><code>ISmsChunkCollection</code></td>
</tr>
<tr>
<td><code>Item</code></td>
<td><code>ISmsChunkCollection</code></td>
</tr>
<tr>
<td><code>Remove</code></td>
<td><code>ISmsChunkCollection</code> ([in] long Index)</td>
</tr>
<tr>
<td><code>RemoveAll</code></td>
<td><code>ISmsChunkCollection</code></td>
</tr>
</tbody>
</table>

Copyright © 2006 Skype Limited. All rights reserved.
Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
# ISmsMessage Member List

This is the complete list of members for ISmsMessage, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>ISmsMessage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Body</td>
<td>ISmsMessage</td>
</tr>
<tr>
<td>Chunks</td>
<td>ISmsMessage</td>
</tr>
<tr>
<td>Delete()</td>
<td>ISmsMessage</td>
</tr>
<tr>
<td>FailureReason</td>
<td>ISmsMessage</td>
</tr>
<tr>
<td>Id</td>
<td>ISmsMessage</td>
</tr>
<tr>
<td>IsFailedUnseen</td>
<td>ISmsMessage</td>
</tr>
<tr>
<td>Price</td>
<td>ISmsMessage</td>
</tr>
<tr>
<td>PriceCurrency</td>
<td>ISmsMessage</td>
</tr>
<tr>
<td>PricePrecision</td>
<td>ISmsMessage</td>
</tr>
<tr>
<td>ReplyToNumber</td>
<td>ISmsMessage</td>
</tr>
<tr>
<td>Seen</td>
<td>ISmsMessage</td>
</tr>
<tr>
<td>Send()</td>
<td>ISmsMessage</td>
</tr>
<tr>
<td>Status</td>
<td>ISmsMessage</td>
</tr>
<tr>
<td>TargetNumbers</td>
<td>ISmsMessage</td>
</tr>
<tr>
<td>Targets</td>
<td>ISmsMessage</td>
</tr>
<tr>
<td>Timestamp</td>
<td>ISmsMessage</td>
</tr>
<tr>
<td>Type</td>
<td>ISmsMessage</td>
</tr>
</tbody>
</table>

Copyright © 2006 Skype Limited. All rights reserved.
<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## ISmsMessageCollection Member List

This is the complete list of members for **ISmsMessageCollection**, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>_NewEnum</td>
<td>ISmsMessageCollection</td>
</tr>
<tr>
<td>Add([in] ISmsMessage *pItem)</td>
<td>ISmsMessageCollection</td>
</tr>
<tr>
<td>Count</td>
<td>ISmsMessageCollection</td>
</tr>
<tr>
<td>Item</td>
<td>ISmsMessageCollection</td>
</tr>
<tr>
<td>Remove([in] long Index)</td>
<td>ISmsMessageCollection</td>
</tr>
<tr>
<td>RemoveAll(void)</td>
<td>ISmsMessageCollection</td>
</tr>
</tbody>
</table>

Copyright © 2006 Skype Limited. All rights reserved.
Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td>Class List</td>
<td>Class Members</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
**ISmsTarget Member List**

This is the complete list of members for **ISmsTarget**, including all inherited members.

<table>
<thead>
<tr>
<th>Message ISmsTarget</th>
<th>Number ISmsTarget</th>
<th>Status ISmsTarget</th>
</tr>
</thead>
</table>

Copyright © 2006 Skype Limited. All rights reserved.
# Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
ISmsTargetCollection Member List

This is the complete list of members for ISmsTargetCollection, including all inherited members.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>_NewEnum</td>
<td>ISmsTargetCollection</td>
</tr>
<tr>
<td>Add([in] ISmsTarget *pItem)</td>
<td>ISmsTargetCollection</td>
</tr>
<tr>
<td>Count</td>
<td>ISmsTargetCollection</td>
</tr>
<tr>
<td>Item</td>
<td>ISmsTargetCollection</td>
</tr>
<tr>
<td>Remove([in] long Index)</td>
<td>ISmsTargetCollection</td>
</tr>
<tr>
<td>RemoveAll(void)</td>
<td>ISmsTargetCollection</td>
</tr>
</tbody>
</table>

Copyright © 2006 Skype Limited. All rights reserved.
# IUser Member List

This is the complete list of members for **IUser**, including all inherited members.

<table>
<thead>
<tr>
<th>Alias</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>About</td>
<td>IUser</td>
</tr>
<tr>
<td>Aliases</td>
<td>IUser</td>
</tr>
<tr>
<td>Birthday</td>
<td>IUser</td>
</tr>
<tr>
<td>BuddyStatus</td>
<td>IUser</td>
</tr>
<tr>
<td>CanLeaveVoicemail</td>
<td>IUser</td>
</tr>
<tr>
<td>City</td>
<td>IUser</td>
</tr>
<tr>
<td>Country</td>
<td>IUser</td>
</tr>
<tr>
<td>CountryCode</td>
<td>IUser</td>
</tr>
<tr>
<td>DisplayName</td>
<td>IUser</td>
</tr>
<tr>
<td>FullName</td>
<td>IUser</td>
</tr>
<tr>
<td>Handle</td>
<td>IUser</td>
</tr>
<tr>
<td>HasCallEquipment</td>
<td>IUser</td>
</tr>
<tr>
<td>Homepage</td>
<td>IUser</td>
</tr>
<tr>
<td>IsAuthorized</td>
<td>IUser</td>
</tr>
<tr>
<td>IsBlocked</td>
<td>IUser</td>
</tr>
<tr>
<td>IsCallForwardActive</td>
<td>IUser</td>
</tr>
<tr>
<td>IsSkypeOutContact</td>
<td>IUser</td>
</tr>
<tr>
<td>IsVideoCapable</td>
<td>IUser</td>
</tr>
<tr>
<td>IsVoicemailCapable</td>
<td>IUser</td>
</tr>
<tr>
<td>Language</td>
<td>IUser</td>
</tr>
<tr>
<td>LanguageCode</td>
<td>IUser</td>
</tr>
<tr>
<td>LastOnline</td>
<td>IUser</td>
</tr>
<tr>
<td>MoodText</td>
<td>IUser</td>
</tr>
<tr>
<td>NumberOfAuthBuddies</td>
<td>IUser</td>
</tr>
<tr>
<td>OnlineStatus</td>
<td>IUser</td>
</tr>
<tr>
<td>PhoneHome</td>
<td>IUser</td>
</tr>
<tr>
<td>PhoneMobile</td>
<td>IUser</td>
</tr>
<tr>
<td>PhoneOffice</td>
<td>IUser</td>
</tr>
<tr>
<td>Province</td>
<td>IUser</td>
</tr>
<tr>
<td>---------------</td>
<td>-------</td>
</tr>
<tr>
<td>ReceivedAuthRequest</td>
<td>IUser</td>
</tr>
<tr>
<td>RichMoodText</td>
<td>IUser</td>
</tr>
<tr>
<td>Sex</td>
<td>IUser</td>
</tr>
<tr>
<td>SpeedDial</td>
<td>IUser</td>
</tr>
<tr>
<td>Timezone</td>
<td>IUser</td>
</tr>
</tbody>
</table>

Copyright © 2006 Skype Limited. All rights reserved.
Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Files</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class List</td>
<td>Class Members</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
IUserCollection Member List

This is the complete list of members for **IUserCollection**, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>_NewEnum</td>
<td>IUserCollection</td>
</tr>
<tr>
<td>Add([in] IUser *pUser)</td>
<td>IUserCollection</td>
</tr>
<tr>
<td>Count</td>
<td>IUserCollection</td>
</tr>
<tr>
<td>Item</td>
<td>IUserCollection</td>
</tr>
<tr>
<td>Remove([in] long Index)</td>
<td>IUserCollection</td>
</tr>
<tr>
<td>RemoveAll(void)</td>
<td>IUserCollection</td>
</tr>
</tbody>
</table>

Copyright © 2006 Skype Limited. All rights reserved.
Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
# IVoicemail Member List

This is the complete list of members for **IVoicemail**, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AllowedDuration</td>
<td>IVoicemail</td>
</tr>
<tr>
<td>Delete()</td>
<td>IVoicemail</td>
</tr>
<tr>
<td>Download()</td>
<td>IVoicemail</td>
</tr>
<tr>
<td>Duration</td>
<td>IVoicemail</td>
</tr>
<tr>
<td>FailureReason</td>
<td>IVoicemail</td>
</tr>
<tr>
<td>Id</td>
<td>IVoicemail</td>
</tr>
<tr>
<td>Open()</td>
<td>IVoicemail</td>
</tr>
<tr>
<td>PartnerDisplayName</td>
<td>IVoicemail</td>
</tr>
<tr>
<td>PartnerHandle</td>
<td>IVoicemail</td>
</tr>
<tr>
<td>SetUnplayed()</td>
<td>IVoicemail</td>
</tr>
<tr>
<td>StartPlayback()</td>
<td>IVoicemail</td>
</tr>
<tr>
<td>StartPlaybackInCall()</td>
<td>IVoicemail</td>
</tr>
<tr>
<td>StartRecording()</td>
<td>IVoicemail</td>
</tr>
<tr>
<td>Status</td>
<td>IVoicemail</td>
</tr>
<tr>
<td>StopPlayback()</td>
<td>IVoicemail</td>
</tr>
<tr>
<td>StopRecording()</td>
<td>IVoicemail</td>
</tr>
<tr>
<td>Timestamp</td>
<td>IVoicemail</td>
</tr>
<tr>
<td>Type</td>
<td>IVoicemail</td>
</tr>
<tr>
<td>Upload()</td>
<td>IVoicemail</td>
</tr>
</tbody>
</table>

Copyright © 2006 Skype Limited. All rights reserved.
## Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
# IVoicemailCollection Member List

This is the complete list of members for **IVoicemailCollection**, including all inherited members.

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>_NewEnum</td>
<td></td>
<td>IVoicemailCollection</td>
</tr>
<tr>
<td>Add([in] IVoicemail *pltem)</td>
<td></td>
<td>IVoicemailCollection</td>
</tr>
<tr>
<td>Count</td>
<td></td>
<td>IVoicemailCollection</td>
</tr>
<tr>
<td>Item</td>
<td></td>
<td>IVoicemailCollection</td>
</tr>
<tr>
<td>Remove([in] long Index)</td>
<td></td>
<td>IVoicemailCollection</td>
</tr>
<tr>
<td>RemoveAll(void)</td>
<td></td>
<td>IVoicemailCollection</td>
</tr>
</tbody>
</table>

Copyright © 2006 Skype Limited. All rights reserved.
Accept() : IGroup
AcceptAdd() : IChat
Add() : IParticipantCollection, IConferenceCollection, ICallChannelCollection, IVoicemailCollection, ICallCollection, IApplicationStreamCollection, IGroupCollection, IUserCollection, IChatMessageCollection, ISmsChunkCollection, ISmsTargetCollection, IChatCollection, ISmsMessageCollection, IFileTransferCollection
AddMembers() : IChat
AddUser() : IGroup
Answer() : ICall
ApplicationConnecting() : _ISkypeEvents
ApplicationDatagram() : _ISkypeEvents
ApplicationReceiving() : _ISkypeEvents
ApplicationSending() : _ISkypeEvents
ApplicationStreams() : _ISkypeEvents
AsyncSearchUsers() : ISkype
AsyncSearchUsersFinished() : _ISkypeEvents
Attach() : ISkype
AttachmentStatus() : _ISkypeEvents
AttachmentStatusToText() : IConversion
AutoAway() : _ISkypeEvents
NEWEnum : IUserCollection, ICallChannelCollection, IFileTransferCollection, ISmsMessageCollection, ISmsTargetCollection, ISmsChunkCollection, IGroupCollection, IApplicationStreamCollection, IVoicemailCollection, IConferenceCollection, IChatCollection, IChatMemberCollection, IChatMessageCollection, ICallCollection, IParticipantCollection

Copyright © 2006 Skype Limited. All rights reserved.
Here is a list of all documented class members with links to the class documentation for each member:

**- a -**

- About: [IUser](#), [IPhone](#)
- Accept(): [IGroup](#)
- AcceptAdd(): [IChat](#)
- ActiveCalls: [IConference](#), [ISkype](#)
- ActiveChats: [ISkype](#)
- ActiveFileTransfers: [ISkype](#)
- ActiveMembers: [IChat](#)
- ActivityTimestamp: [IChat](#)
- Add(): [IConferenceCollection](#), [IVoicemailCollection](#), [IApplicationStreamCollection](#), [IGroupCollection](#), [ISmsChunkCollection](#), [ISmsTargetCollection](#), [IUserCollection](#), [ISmsMessageCollection](#), [IFileTransferCollection](#), [IParticipantCollection](#), [ICallChannelCollection](#), [ICallCollection](#), [ICallMessageCollection](#), [IChatCollection](#)
- Adder: [IChat](#)
- AddMembers(): [IChat](#)
- AddUser(): [IGroup](#)
- AEC: [ISettings](#)
- AGC: [ISettings](#)
- AlertString: [IChat](#)
- Aliases: [IUser](#)
- AllowedDuration: [IVoicemail](#)
• Answer() : ICall
• ApiSecurityContextEnabled : ISkype
• ApiWrapperVersion : ISkype
• Applicants : IChat
• Application : ISkype
• ApplicationConnecting() : _ISkypeEvents
• ApplicationDatagram() : _ISkypeEvents
• ApplicationName : IApplicationStream
• ApplicationReceiving() : _ISkypeEvents
• ApplicationSending() : _ISkypeEvents
• ApplicationStreams() : _ISkypeEvents
• AsyncSearchUsers() : ISkype
• AsyncSearchUsersFinished() : _ISkypeEvents
• Attach() : ISkype
• AttachmentStatus : ISkype , _ISkypeEvents
• AttachmentStatusToText() : IConversion
• AudioIn : ISettings
• AudioOut : ISettings
• AutoAway : ISettings , _ISkypeEvents
• Avatar : ISettings

Copyright © 2006 Skype Limited. All rights reserved.
Here is a list of all documented class members with links to the class documentation for each member:

- **b** -
  
  - Balance : **IProfile**
  - BalanceCurrency : **IProfile**
  - BalanceToText : **IProfile**
  - Birthday : **IUser**, **IProfile**
  - Blob : **IChat**
  - Blocking : **ICommand**
  - Body : **IChatMessage**, **ISmsMessage**
  - Bookmark() : **IChat**
  - Bookmarked : **IChat**
  - BookmarkedChats : **ISkype**
  - BuddyStatus : **IUser**
  - BuddyStatusToText() : **IConversion**
  - ButtonPressed() : **IClient**
  - ButtonReleased() : **IClient**
  - BytesPerSecond : **IFileTransfer**
  - BytesTransferred : **IFileTransfer**

Copyright © 2006 Skype Limited. All rights reserved.
Here is a list of all documented class members with links to the class documentation for each member:

- c -

- Cache : ISkype
- Call : ISkype, ICallChannel
- CallApplyCF : IProfile
- CallDtmfReceived() : _ISkypeEvents
- CallFailureReasonToText() : IConversion
- CallForwardRules : IProfile
- CallHistory() : _ISkypeEvents
- CallInputStatusChanged() : _ISkypeEvents
- CallNoAnswerTimeout : IProfile
- Calls : IConference, ISkype
- CallSeenStatusChanged() : _ISkypeEvents
- CallSendToVM : IProfile
- CallStatus() : _ISkypeEvents, IParticipant
- CallStatusToText() : IConversion
- CallTransferStatusChanged() : _ISkypeEvents
- CallType : IParticipant
- CallTypeToText() : IConversion
- CallVideoReceiveStatusChanged() : _ISkypeEvents
- CallVideoSendStatusChanged() : _ISkypeEvents
- CallVideoSendStatusToText() : IConversion
- CallVideoStatusChanged() : _ISkypeEvents
- CallVideoStatusToText() : IConversion
- CanLeaveVoicemail : IUser
- CanSetRoleTo: IChatMember
- CanTransfer: ICall
- Caption: IPluginMenuItem
- CaptureMicDevice: ICall
- ChangeUserStatus(): ISkype
- Channels: ICallChannelManager, ICallChannelManagerEvents
- ChannelType: ICallChannelManager
- CharactersLeft: ISmsChunk
- Chat: IChatMessage, IChatMember, ISkype
- ChatLeaveReasonToText(): IConversion
- ChatMemberRoleChanged(): _ISkypeEvents
- ChatMembersChanged(): _ISkypeEvents
- ChatMessageStatusToText(): IConversion
- ChatMessageTypeToText(): IConversion
- ChatName: IChatMessage
- Chats: ISkype
- ChatStatusToText(): IConversion
- Chunks: ISmsMessage
- City: IUser, IProfile
- ClearCallHistory(): ISkype
- ClearChatHistory(): ISkype
- ClearRecentMessages(): IChat
- ClearVoicemailHistory(): ISkype
- Client: ISkype
- Command: ICommand, ISkype, _ISkypeEvents
- CommandId: ISkype
- Conference: ISkype
- ConferenceId: ICall
- Conferences: ISkype
- Connect(): ICallChannelManager, IApplication
- ConnectableUsers: IApplication
- ConnectingUsers: IApplication
- ConnectionStatus: ISkype, _ISkypeEvents
- ConnectionStatusToText(): IConversion
- ContactsFocused(): _ISkypeEvents
- Convert: ISkype
- Count: IChatMessageCollection, IChatCollection, IFileTransferCollection, ICallCollection,
IParticipantCollection, IUserCollection, ICallChannelCollection, IGroupCollection, ISmsTargetCollection, IChatMemberCollection, IVoicemailCollection, IApplicationStreamCollection, ISmsChunkCollection, ISmsMessageCollection, IConferenceCollection

- Country : IUser, IProfile
- CountryCode : IUser
- Create() : IApplication
- CreateApplication() : ICallChannelManager
- CreateChatMultiple() : ISkype
- CreateChatUsingBlob() : ISkype
- CreateChatWith() : ISkype
- Created : ICallChannelManager, _ICallChannelManagerEvents
- CreateEvent() : IClient
- CreateGroup() : ISkype
- CreateMenuItem() : IClient
- CreateSms() : ISkype
- CurrentUser : ISkype
- CurrentUserHandle : ISkype
- CurrentUserProfile : ISkype
- CurrentUserStatus : ISkype
- CustomGroupId : IGroup
- CustomGroups : ISkype

Copyright © 2006 Skype Limited. All rights reserved.
Here is a list of all documented class members with links to the class documentation for each member:

- d -

- DataLength : IApplicationStream
- Decline() : IGroup
- Delete() : IApplication,ISmsMessage,IVoicemail, IPluginMenuItem, IPluginEvent
- DeleteGroup() : ISkype
- Description : IChat
- DialogPartner : IChat
- Disband() : IChat
- Disconnect() : IApplicationStream,ICallChannelManager
- DisplayName : IGroup, IParticipant, IUser
- Download() : IVoicemail
- DTMF : ICall
- Duration : ICall, IVoicemail

Copyright ✶ 2006 Skype Limited. All rights reserved.
Here is a list of all documented class members with links to the class documentation for each member:

- e -

- EditedBy : IChatMessage
- EditedTimestamp : IChatMessage
- EnableApiSecurityContext() : ISkype
- Enabled : IPluginMenuItem
- EnterPassword() : IChat
- Error() : _ISkypeEvents
- Expected : ICommand
Here is a list of all documented class members with links to the class documentation for each member:

- f -

- FailureReason : ICall, IVoicemail, IFileTransfer, ISmsMessage
- FileName : IFileTransfer
- FilePath : IFileTransfer
- FileTransfers : ISkype
- FileTransferStatusChanged() : _ISkypeEvents
- FindChatUsingBlob() : ISkype
- Finish() : ICall, IConference
- FinishTime : IFileTransfer
- Focus() : IClient
- FocusedContacts : ISkype
- Forward() : ICall
- ForwardedBy : ICall
- FriendlyName : ISkype, IChat
- Friends : ISkype
- FromDisplayName : IChatMessage
- FromHandle : IChatMessage
- FullName : IProfile, IUser
Here is a list of all documented class members with links to the class documentation for each member:

- g -

- Greeting : ISkype
- GroupDeleted() : _ISkypeEvents
- GroupExpanded() : _ISkypeEvents
- Groups : ISkype
- GroupTypeToText() : IConversion
- GroupUsers() : _ISkypeEvents
- GroupVisible() : _ISkypeEvents
- GuideLines : IChat

Copyright ⓒ 2006 Skype Limited. All rights reserved.
Here is a list of all documented class members with links to the class documentation for each member:

- **h** -

  - Handle: **IUser**, **IParticipant**, **IApplicationStream**, **IChatMember**
  - HardwiredGroups: **ISkype**
  - HasCallEquipment: **IUser**
  - Hint: **IPluginMenuItem**
  - Hold(): **ICall**, **IConference**
  - Homepage: **IProfile**, **IUser**

Copyright © 2006 Skype Limited. All rights reserved.
Here is a list of all documented class members with links to the class documentation for each member:

- i -

- Id: ICommand, ICall, IChatMember, ISmsChunk, ISmsMessage, IConference, IPluginMenuitem, IPluginEvent, IChatMessage, IVoicemail, IFileTransfer, IGroup
- InputDevice: ICall
- InputStatus: ICall
- IPCountry: IProfile
- IsActive: IChatMember
- IsAuthorized: IUser
- IsBlocked: IUser
- IsCallForwardActive: IUser
- IsEditable: IChatMessage
- IsExpanded: IGroup
- IsFailedUnseen: ISmsMessage
- IsRunning: IClient
- IsSkypeOutContact: IUser
- IsVideoCapable: IUser
- IsVisible: IGroup
- IsVoicemailCapable: IUser
- Item: ICallChannelCollection, IChatMessageCollection, IChatCollection, ISmsTargetCollection, IParticipantCollection, IVoicemailCollection, IChatMemberCollection, IConferenceCollection, IApplicationStreamCollection, ICallCollection, IUserCollection, ISmsChunkCollection,
IGroupCollection, IFileTransferCollection, ISmsMessageCollection

Copyright � 2006 Skype Limited. All rights reserved.
Here is a list of all documented class members with links to the class documentation for each member:

- j -

  - Join() : **ICall**, **IChat**

---

Copyright ✨ 2006 **Skype Limited**. All rights reserved.
Here is a list of all documented class members with links to the class documentation for each member:

- **k** -
  
  - Kick() : [IChat](#)
  - KickBan() : [IChat](#)
Here is a list of all documented class members with links to the class documentation for each member:

- I -

- Language : IUserInfo, IConversion, ISettings
- LanguageCode : IUserInfo
- Languages : IProfile
- LastOnline : IUserInfo
- Leave() : IChat
- LeaveReason : IChatMessage
- LoadAvatarFromFile() : ISettings
Here is a list of all documented class members with links to the class documentation for each member:

- m -

- Manager : ICallChannel
- MemberObjects : IChat
- Members : IChat
- Message : ISmsChunk, ISmsTarget, ISkype, ICallChannelManagerEvents
- MessageHistory() : _ISkypeEvents
- Messages : IChat, ISkype
- MessageStatus() : _ISkypeEvents
- Minimize() : IClient
- MissedCalls : ISkype
- MissedChats : ISkype
- MissedMessages : ISkype
- MissedSmss : ISkype
- MissedVoicemails : ISkype
- MoodText : IUser, IProfile
- Mute() : _ISkypeEvents, ISkype
- MyRole : IChat
- MyStatus : IChat
Here is a list of all documented class members with links to the class documentation for each member:

- n -

- Name: IChat, IApplication, ICallChannelManager
- Number: ISmsTarget
- NumberOfAuthBuddies: IUser

Copyright Ⓒ 2006 Skype Limited. All rights reserved.
Here is a list of all documented class members with links to the class documentation for each member:

- o -

- OnlineStatus : IUser , _ISkypeEvents
- OnlineStatusToText() : IConversion
- OnlineUsers : IGroup
- Open() : IVoicemail
- OpenAddContactDialog() : IClient
- OpenAuthorizationDialog() : IClient
- OpenBlockedUsersDialog() : IClient
- OpenCallHistoryTab() : IClient
- OpenConferenceDialog() : IClient
- OpenContactsTab() : IClient
- OpenDialog() : IClient
- OpenDialpadTab() : IClient
- OpenFileTransferDialog() : IClient
- OpenGettingStartedWizard() : IClient
- OpenImportContactsWizard() : IClient
- OpenLiveTab() : IClient
- OpenMessageDialog() : IClient
- OpenOptionsDialog() : IClient
- OpenProfileDialog() : IClient
- OpenSearchDialog() : IClient
- OpenSendContactsDialog() : IClient
- OpenSmsDialog() : IClient
- OpenUserInfoDialog() : IClient
- OpenVideoTestDialog() : IClient
- OpenWindow() : IChat
- Options : IChatMessage , IChat
- OutputDevice : ICall
Here is a list of all documented class members with links to the class documentation for each member:

- **p** -

- Participants : `ICall`
- PartnerDisplayName : `ICall`, `IFileTransfer`, `IVoicemail`
- PartnerHandle : `IFileTransfer`, `ICall`, `IVoicemail`, `IApplicationStream`
- PasswordHint : `IChat`
- PCSpeaker : `ISettings`
- PhoneHome : `IUser`, `IProfile`
- PhoneMobile : `IUser`, `IProfile`
- PhoneOffice : `IProfile`, `IUser`
- PlaceCall() : `ISkype`
- PluginEventClicked() : `_ISkypeEvents`
- PluginMenuItemClicked() : `_ISkypeEvents`
- Posters : `IChat`
- PredictiveDialerCountry : `ISkype`
- Price : `ISmsMessage`
- PriceCurrency : `ISmsMessage`
- PricePrecision : `ISmsMessage`
- Privilege : `ISkype`
- Profile : `ISkype`
- Property : `ISkype`
- Protocol : `ISkype`
- Province : `IUser`, `IProfile`
- PstnNumber : `ICall`
PstnStatus : ICall
Here is a list of all documented class members with links to the class documentation for each member:

- r -

- Rate : ICall
- RateCurrency : ICall
- RatePrecision : ICall
- Read() : IApplicationStream
- ReceivedAuthRequest : IUser
- ReceivedStreams : IApplication
- RecentChats : ISkype
- RecentMessages : IChat
- RedirectToVoicemail() : ICall
- Remove() : IVoicemailCollection, IApplicationStreamCollection, IGroupCollection, ISmsChunkCollection, IUserCollection, ISmsTargetCollection, ISmsMessageCollection, IParticipantCollection, IFileTransferCollection, ICallChannelCollection, ICallCollection, IChatMessageCollection, IChatCollection, IConferenceCollection
- RemoveAll() : ISmsMessageCollection, IGroupCollection, ISmsChunkCollection, IChatMessageCollection, ISmsTargetCollection, IFileTransferCollection, ICallChannelCollection, IApplicationStreamCollection, IUserCollection, IParticipantCollection, ICallCollection, IChatCollection, IConferenceCollection, IVoicemailCollection
- RemoveUser() : IGroup
- Reply() : _ISkypeEvents, ICommand
- ReplyToNumber : ISmsMessage
- ResetCache() : ISkype
- ResetIdleTimer() : ISettings
- Resume() : IConference, ICall
- RichMoodText : IUser, IProfile
- Ringer : ISettings
- RingTone : ISettings
- RingToneStatus : ISettings
- Role : IChatMessage, IChatMember
Here is a list of all documented class members with links to the class documentation for each member:

- s -

- SaveAvatarToFile() : ISettings
- SearchForUsers() : ISkype
- Seen : IChatMessage, ISmsMessage, ICall
- Send() : ISmsMessage
- SendCommand() : ISkype
- SendDatagram() : IApplicationStream, IApplication
- Sender : IChatMessage
- SendingStreams : IApplication
- SendMessage() : IChat, ISkype
- SendSm() : ISkype
- SendTextMessage() : ICallChannel
- SendVoicemail() : ISkype
- SetPassword() : IChat
- Settings : ISkype
- SetUnplayed() : IVoicemail
- Sex : IUser, IProfile
- Share() : IGroup
- Shutdown() : IClient
- SilentMode : ISkype
- SilentModeStatusChanged() : _ISkypeEvents
- SmsMessageStatusChanged() : _ISkypeEvents
- SmsMessageStatusToText() : IConversion
- SmsMessageTypeToText() : IConversion
• Smss : ISkype
• SmsTargetStatusChanged() : _ISkypeEvents
• SmsTargetStatusToText() : IConversion
• SpeedDial : IUser
• Start() : IClient
• StartPlayback() : IVoicemail
• StartPlaybackInCall() : IVoicemail
• StartRecording() : IVoicemail
• StartTime : IFileTransfer
• StartVideoReceive() : ICall
• StartVideoSend() : ICall
• Status : IFileTransfer , ISmsMessage , ICall , ISmsTarget , IVoicemail , IChatMessage , IChat
• StopPlayback() : IVoicemail
• StopRecording() : IVoicemail
• StopVideoReceive() : ICall
• StopVideoSend() : ICall
• Stream : ICallChannel
• Streams : IApplication
• Subject : ICall

Copyright ¤ 2006 Skype Limited. All rights reserved.
Here is a list of all documented class members with links to the class documentation for each member:

- t -

- TargetIdentity : ICall
- TargetNumbers : ISmsMessage
- Targets : ISmsMessage
- Text : ISmsChunk, ICallChannelMessage
- TextToAttachmentStatus() : IConversion
- TextToBuddyStatus() : IConversion
- TextToCallStatus() : IConversion
- TextToCallType() : IConversion
- TextToChatMessageStatus() : IConversion
- TextToChatMessageType() : IConversion
- TextToConnectionStatus() : IConversion
- TextToGroupType() : IConversion
- TextToOnlineStatus() : IConversion
- TextToUserSex() : IConversion
- TextToUserStatus() : IConversion
- TextToVoicemailStatus() : IConversion
- Timeout : ICommand, ISkype
- Timestamp : IVoicemail, ICall, IChatMessage, IChat, ISmsMessage
- Timezone : IUser, IProfile
- Topic : IChat
- TopicXML : IChat
- Transfer() : ICall
- TransferActive: ICall
- TransferredBy: ICall
- TransferredTo: ICall
- TransferStatus: ICall
- Type: IVoicemail, ISmsMessage, IChatMessage, IChat, ICall, ICallChannel, IGroup, IFileTransfer
Here is a list of all documented class members with links to the class documentation for each member:

- **u** -

  - UILanguageChanged() : _ISkypeEvents
  - Unbookmark() : IChat
  - Upload() : IVoicemail
  - User : ISkype
  - UserAuthorizationRequestReceived() : _ISkypeEvents
  - UserMood() : _ISkypeEvents
  - Users : IGroup, IChatMessage
  - UserSexToText() : IConversion
  - UserStatus() : _ISkypeEvents
  - UserStatusToText() : IConversion
  - UsersWaitingAuthorization : ISkype
Here is a list of all documented class members with links to the class documentation for each member:

- v -

- ValidatedSmsNumbers : IProfile
- Variable : ISkype
- Version : ISkype
- VideoIn : ISettings
- VideoReceiveStatus : ICall
- VideoSendStatus : ICall
- VideoStatus : ICall
- VmAllowedDuration : ICall
- VmDuration : ICall
- Voicemail : ISkype
- VoicemailFailureReasonToText() : IConversion
- Voicemails : ISkype
- VoicemailStatus() : _ISkypeEvents
- VoicemailStatusToText() : IConversion
- VoicemailTypeToText() : IConversion

Copyright © 2006 Skype Limited. All rights reserved.
Here is a list of all documented class members with links to the class documentation for each member:

- **W** -
  
  - Wallpaper : **IClient**
  - WallpaperChanged() : **ISkypeEvents**
  - Write() : **IApplicationStream**
- b -

- Bookmark() : **IChat**
- BuddyStatusToText() : **IConversion**
- ButtonPressed() : **IClient**
- ButtonReleased() : **IClient**

Copyright � 2006 Skype Limited. All rights reserved.
### Skype4COM 1.0.36.0

#### Files | Examples
--- | ---
Class List | Class Members

#### Class Members

- **a**
  - CallDtmfReceived() : _ISkypeEvents
  - CallFailureReasonToText() : IConversion
  - CallHistory() : _ISkypeEvents
- **b**
  - CallInputStatusChanged() : _ISkypeEvents
  - CallSeenStatusChanged() : _ISkypeEvents
  - CallStatus() : _ISkypeEvents
- **c**
  - CallStatusToText() : IConversion
  - CallTransferStatusChanged() : _ISkypeEvents
  - CallTypeToText() : IConversion
  - CallVideoReceiveStatusChanged() : _ISkypeEvents
- **d**
  - CallVideoSendStatusChanged() : _ISkypeEvents
  - CallVideoSendStatusToText() : IConversion
  - CallVideoStatusChanged() : _ISkypeEvents
  - CallVideoStatusToText() : IConversion
- **e**
  - ChangeUserStatus() : ISkype
  - Channels() : _ICallChannelManagerEvents
  - ChatLeaveReasonToText() : IConversion
  - ChatMemberRoleChanged() : _ISkypeEvents
  - ChatMembersChanged() : _ISkypeEvents
- **f**
  - ChatMessageStatusToText() : IConversion
  - ChatMessageTypeToText() : IConversion
  - ChatStatusToText() : IConversion
  - ClearCallHistory() : ISkype
  - ClearChatHistory() : ISkype
• ClearRecentMessages(): IChat
• ClearVoicemailHistory(): ISkype
• Command(): _ISkypeEvents
• Connect(): IApplication, ICallChannelManager
• ConnectionStatus(): _ISkypeEvents
• ConnectionStatusToText(): IConversion
• ContactsFocused(): _ISkypeEvents
• Create(): IApplication
• CreateApplication(): ICallChannelManager
• CreateChatMultiple(): ISkype
• CreateChatUsingBlob(): ISkype
• CreateChatWith(): ISkype
• Created(): _ICallChannelManagerEvents
• CreateEvent(): IClient
• CreateGroup(): ISkype
• CreateMenuItem(): IClient
• CreateSms(): ISkype

Copyright © 2006 Skype Limited. All rights reserved.
Skype4COM 1.0.36.0

- d -

- Decline() : IGroup
- Delete() : IVoicemail, ISmsMessage, IPluginMenuItem, IApplication, IPluginEvent
- DeleteGroup() : ISkype
- Disband() : IChat
- Disconnect() : IApplicationStream, ICallChannelManager
- Download() : IVoicemail

Copyright � 2006 Skype Limited. All rights reserved.
<table>
<thead>
<tr>
<th>Class Members</th>
<th>Functions</th>
<th>Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>EnableApiSecurityContext()</td>
<td>ISkype</td>
<td></td>
</tr>
<tr>
<td>EnterPassword()</td>
<td>IChat</td>
<td></td>
</tr>
<tr>
<td>Error()</td>
<td>_ISkypeEvents</td>
<td></td>
</tr>
</tbody>
</table>

Copyright © 2006 Skype Limited. All rights reserved.
- f -

- FileTransferStatusChanged() : _ISkypeEvents
- FindChatUsingBlob() : ISkype
- Finish() : IConference, ICall
- Focus() : IClient
- Forward() : ICall

Copyright © 2006 Skype Limited. All rights reserved.
- g -

- GroupDeleted() : _ISkypeEvents
- GroupExpanded() : _ISkypeEvents
- GroupTypeToText() : IConversion
- GroupUsers() : _ISkypeEvents
- GroupVisible() : _ISkypeEvents

Copyright ✩ 2006 Skype Limited. All rights reserved.
- h -

- Hold() : ICall, IConference

Copyright © 2006 Skype Limited. All rights reserved.
- j -

- Join() : ICall, IChat

Copyright © 2006 Skype Limited. All rights reserved.
- k -

- Kick() : IChat
- KickBan() : IChat
### Skype4COM 1.0.36.0

**Main Page** | **Related Pages** | **Modules** | **Classes**
---|---|---|---
**Files** | **Examples** |
**Class List** | **Class Members** |
**All** | **Functions** | **Properties** |

| a | b | c | d | e | f | g | h | j | k | l | m | o | p | r | s | t | u | v | w |

### - I -

- Leave() : **IChat**
- LoadAvatarFromFile() : **ISettings**

---

Copyright ♦ 2006 Skype Limited. All rights reserved.
Skype4COM 1.0.36.0

- m -

- Message() : _ICallChannelManagerEvents
- MessageHistory() : _ISkypeEvents
- MessageStatus() : _ISkypeEvents
- Minimize() : IClient
- Mute() : _ISkypeEvents

Copyright © 2006 Skype Limited. All rights reserved.
- o -

- OnlineStatus() : _ISkypeEvents
- OnlineStatusToText() : IConversion
- Open() : IVoicemail
- OpenAddContactDialog() : IClient
- OpenAuthorizationDialog() : IClient
- OpenBlockedUsersDialog() : IClient
- OpenCallHistoryTab() : IClient
- OpenConferenceDialog() : IClient
- OpenContactsTab() : IClient
- OpenDialog() : IClient
- OpenDialpadTab() : IClient
- OpenFileTransferDialog() : IClient
- OpenGettingStartedWizard() : IClient
- OpenImportContactsWizard() : IClient
- OpenLiveTab() : IClient
- OpenMessageDialog() : IClient
- OpenOptionsDialog() : IClient
- OpenProfileDialog() : IClient
- OpenSearchDialog() : IClient
- OpenSendContactsDialog() : IClient
- OpenSmsDialog() : IClient
- OpenUserInfoDialog() : IClient
- OpenVideoTestDialog() : IClient
- OpenWindow() : IChat
Copyright © 2006 Skype Limited. All rights reserved.
```
- p -

- PlaceCall() : ISkype
- PluginEventClicked() : _ISkypeEvents
- PluginMenuItemClicked() : _ISkypeEvents
```

---

Copyright © 2006 Skype Limited. All rights reserved.
- r -

- Read() : IApplicationStream
- RedirectToVoicemail() : ICall
- Remove() : IParticipantCollection, IConferenceCollection, ICallChannelCollection, IVoicemailCollection, ICallCollection, IApplicationStreamCollection, IGroupCollection, IUserCollection, IChatMessageCollection, ISmsChunkCollection, ISmsTargetCollection, IChatCollection, ISmsMessageCollection, IFileTransferCollection
- RemoveAll() : IChatCollection, IFileTransferCollection, IUserCollection, IParticipantCollection, ICallCollection, ICallChannelCollection, IApplicationStreamCollection, IGroupCollection, ISmsTargetCollection, IChatMessageCollection, IConferenceCollection, ISmsChunkCollection, ISmsMessageCollection, IVoicemailCollection
- RemoveUser() : IGroup
- Reply() : _ISkypeEvents
- ResetCache() : ISkype
- ResetIdleTimer() : ISettings
- Resume() : ICall, IConference
- s -

- SaveAvatarToFile() : ISettings
- SearchForUsers() : ISkype
- Send() : ISmsMessage
- SendCommand() : ISkype
- SendDatagram() : IApplicationStream, IApplication
- SendMessage() : ISkype, IChat
- SendSms() : ISkype
- SendTextMessage() : ICallChannel
- SendVoicemail() : ISkype
- SetPassword() : IChat
- SetUnplayed() : IVoicemail
- Share() : IGroup
- Shutdown() : IClient
- SilentModeStatusChanged() : _ISkypeEvents
- SmsMessageStatusChanged() : _ISkypeEvents
- SmsMessageStatusToText() : IConversion
- SmsMessageTypeToText() : IConversion
- SmsTargetStatusChanged() : _ISkypeEvents
- SmsTargetStatusToText() : IConversion
- Start() : IClient
- StartPlayback() : IVoicemail
- StartPlaybackInCall() : IVoicemail
- StartRecording() : IVoicemail
- StartVideoReceive() : ICall
• StartVideoSend() : ICall
• StopPlayback() : IVoicemail
• StopRecording() : IVoicemail
• StopVideoReceive() : ICall
• StopVideoSend() : ICall
- t -

- TextToAttachmentStatus() : IConversion
- TextToBuddyStatus() : IConversion
- TextToCallStatus() : IConversion
- TextToCallType() : IConversion
- TextToChatMessageStatus() : IConversion
- TextToChatMessageType() : IConversion
- TextToConnectionStatus() : IConversion
- TextToGroupType() : IConversion
- TextToOnlineStatus() : IConversion
- TextToUserSex() : IConversion
- TextToUserStatus() : IConversion
- TextToVoicemailStatus() : IConversion
- Transfer() : ICall

Copyright © 2006 Skype Limited. All rights reserved.
Skype4COM 1.0.36.0

- u -

- UILanguageChanged() : _ISkypeEvents
- Unbookmark() : IChat
- Upload() : IVoicemail
- UserAuthorizationRequestReceived() : _ISkypeEvents
- UserMood() : _ISkypeEvents
- UserSexToText() : IConversion
- UserStatus() : _ISkypeEvents
- UserStatusToText() : IConversion

Copyright ♦ 2006 Skype Limited. All rights reserved.


- v -

- VoicemailFailureReasonToText() : `IConversion`
- VoicemailStatus() : `_ISkypeEvents`
- VoicemailStatusToText() : `IConversion`
- VoicemailTypeToText() : `IConversion`

Copyright © 2006 Skype Limited. All rights reserved.
- w -

- WallpaperChanged() : _ISkypeEvents
- Write() : IApplicationStream
- a -

- About: IUser, IProfile
- ActiveCalls: ISkype, IConference
- ActiveChats: ISkype
- ActiveFileTransfers: ISkype
- ActiveMembers: IChat
- ActivityTimestamp: IChat
- Adder: IChat
- AEC: ISettings
- AGC: ISettings
- AlertString: IChat
- Aliases: IUser
- AllowedDuration: IVoicemail
- ApiSecurityContextEnabled: ISkype
- ApiWrapperVersion: ISkype
- Applicants: IChat
- Application: ISkype
- ApplicationName: IApplicationStream
- AttachmentStatus: ISkype
- AudioIn: ISettings
- AudioOut: ISettings
- AutoAway: ISettings
- Avatar: ISettings
- b -

- Balance : IProfile
- BalanceCurrency : IProfile
- BalanceToText : IProfile
- Birthday : IUser , IProfile
- Blob : IChat
- Blocking : ICommand
- Body : IChatMessage , ISmsMessage
- Bookmarked : IChat
- BookmarkedChats : ISkype
- BuddyStatus : IUser
- BytesPerSecond : IFileTransfer
- BytesTransferred : IFileTransfer

Copyright ✪ 2006 Skype Limited. All rights reserved.
- c -

- Cache : ISkype
- Call : ISkype, ICallChannel
- CallApplyCF : IProfile
- CallForwardRules : IProfile
- CallNoAnswerTimeout : IProfile
- Calls : ISkype, IConference
- CallSendToVM : IProfile
- CallStatus : IParticipant
- CallType : IParticipant
- CanLeaveVoicemail : IUser
- CanSetRoleTo : IChatMember
- CanTransfer : ICommand
- Caption : ICommand
- CharactersLeft : ISmsChunk
- Chat : IChatMessage, IChatMember, ISkype
- ChatName : IChatMessage
- Chats : ISkype
- Chunks : ISmsMessage
- City : IUser, IProfile
- Client : ISkype
- Command : ICommand, ISkype
• CommandId : ISkype
• Conference : ISkype
• ConferenceId : ICall
• Conferences : ISkype
• ConnectableUsers : IApplication
• ConnectingUsers : IApplication
• ConnectionStatus : ISkype
• Convert : ISkype
• Count : ICallCollection, ICallChannelCollection, IApplicationStreamCollection, IConferenceCollection, IChatMemberCollection, IVoicemailCollection, IParticipantCollection, IChatMessageCollection, IChatCollection, IUserCollection, ISmsTargetCollection, IGroupCollection, ISmsMessageCollection, IFileTransferCollection, ISmsChunkCollection
• Country : IUser, IProfile
• CountryCode : IUser
• Created : ICallChannelManager
• CurrentUser : ISkype
• CurrentUserHandle : ISkype
• CurrentUserProfile : ISkype
• CurrentUserStatus : ISkype
• CustomGroupId : IGroup
• CustomGroups : ISkype
### Class Members

- **DataLength**: `IApplicationStream`
- **Description**: `IChat`
- **DialogPartner**: `IChat`
- **DisplayName**: `IUser`, `IParticipant`, `IGroup`
- **DTMF**: `ICall`
- **Duration**: `ICall`, `IVoicemail`

Copyright Ⓡ 2006 Skype Limited. All rights reserved.
- e -

- EditedBy : IChatMessage
- EditedTimestamp : IChatMessage
- Enabled : IPluginMenuItem
- Expected : ICommand

Copyright � 2006 Skype Limited. All rights reserved.
FailureReason: ICall, IVoicemail, IFileTransfer, ISmsMessage
FileName: IFileTransfer
FilePath: IFileTransfer
FileTransfers: ISkype
FinishTime: IFileTransfer
FocusedContacts: ISkype
ForwardedBy: ICall
FriendlyName: IChat, ISkype
Friends: ISkype
FromDisplayName: IChatMessage
FromHandle: IChatMessage
FullName: IProfile, IUser
Skype4COM 1.0.36.0

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
<tr>
<td>All</td>
<td>Functions</td>
<td>Properties</td>
<td></td>
</tr>
<tr>
<td>_ a b c d e f g h i l m n o p r s t u v w</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- g -

- Greeting : ISkype
- Groups : ISkype
- GuideLines : IChat

Copyright © 2006 Skype Limited. All rights reserved.
- h -

- Handle : IUser, IParticipant, IApplicationStream, IChatMember
- HardwiredGroups : ISkype
- HasCallEquipment : IUser
- Hint : IPluginMenuItem
- Homepage : IUser, IProfile

Copyright ⚛ 2006 Skype Limited. All rights reserved.
- i -

- Id: ICommand, ICall, IChatMember, ISmsChunk, ISmsMessage, IConference, IPluginMenuitem, IPluginEvent, IChatMessage, IVoicemail, IFileTransfer, IGroup
- InputDevice: ICall
- InputStatus: ICall
- IPCountry: IProfile
- IsActive: IChatMember
- IsAuthorized: IUser
- IsBlocked: IUser
- IsCallForwardActive: IUser
- IsEditable: IChatMessage
- IsExpanded: IGroup
- IsFailedUnseen: ISmsMessage
- IsRunning: IClient
- IsSkypeOutContact: IUser
- IsVideoCapable: IUser
- IsVisible: IGroup
- IsVoicemailCapable: IUser
- Item: ICallChannelCollection, IChatMessageCollection, IChatCollection, ISmsTargetCollection, IParticipantCollection, IVoicemailCollection, IChatMemberCollection, IConferenceCollection, IApplicationStreamCollection, ICallCollection, IUserCollection, ISmsChunkCollection, IGroupCollection, IFileTransferCollection,
ISmsMessageCollection

Copyright © 2006 Skype Limited. All rights reserved.
- I -

- Language : IUser, IConversion, ISettings
- LanguageCode : IUser
- Languages : IProfile
- LastOnline : IUser
- LeaveReason : IChatMessage

Copyright �� 2006 Skype Limited. All rights reserved.
**Skype4COM 1.0.36.0**

<table>
<thead>
<tr>
<th>Main Page</th>
<th>Related Pages</th>
<th>Modules</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>Examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class List</td>
<td>Class Members</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>All</th>
<th>Functions</th>
<th>Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **m** -

- Manager: `ICallChannel`
- MemberObjects: `IChat`
- Members: `IChat`
- Message: `ISmsChunk`, `ISmsTarget`, `ISkype`
- Messages: `ISkype`, `IChat`
- MissedCalls: `ISkype`
- MissedChats: `ISkype`
- MissedMessages: `ISkype`
- MissedSmss: `ISkype`
- MissedVoicemails: `ISkype`
- MoodText: `IProfile`, `IUser`
- Mute: `ISkype`
- MyRole: `IChat`
- MyStatus: `IChat`

---

Copyright © 2006 Skype Limited. All rights reserved.
Skype4COM 1.0.36.0

- n -

- Name : IChat, IApplication, ICallChannelManager
- Number : ISmsTarget
- NumberOfAuthBuddies : IUser

Copyright © 2006 Skype Limited. All rights reserved.
Skype4COM 1.0.36.0

- o -

- OnlineStatus : IUser
- OnlineUsers : IGroup
- Options : IChat, IChatMessage
- OutputDevice : ICall

Copyright � 2006 Skype Limited. All rights reserved.
- p -

- Participants: `ICall`
- PartnerDisplayName: `ICall`, `IFileTransfer`, `IVoicemail`
- PartnerHandle: `IFileTransfer`, `ICall`, `IVoicemail`, `IApplicationStream`
- PasswordHint: `IChat`
- PCSpeaker: `ISettings`
- PhoneHome: `IUser`, `IProfile`
- PhoneMobile: `IUser`, `IProfile`
- PhoneOffice: `IProfile`, `IUser`
- Posters: `IChat`
- PredictiveDialerCountry: `ISkype`
- Price: `ISmsMessage`
- PriceCurrency: `ISmsMessage`
- PricePrecision: `ISmsMessage`
- Privilege: `ISkype`
- Profile: `ISkype`
- Property: `ISkype`
- Protocol: `ISkype`
- Province: `IProfile`, `IUser`
- PstnNumber: `ICall`
- PstnStatus: `ICall`
- r -

- Rate : ICall
- RateCurrency : ICall
- RatePrecision : ICall
- ReceivedAuthRequest : IUser
- ReceivedStreams : IApplication
- RecentChats : ISkype
- RecentMessages : IChat
- Reply : ICommand
- ReplyToNumber : ISmsMessage
- RichMoodText : IUser, IProfile
- Ringer : ISettings
- RingTone : ISettings
- RingToneStatus : ISettings
- Role : IChatMessage, IChatMember
- s -

- Seen : ICall, IChatMessage, ISmsMessage
- Sender : IChatMessage
- SendingStreams : IApplication
- Settings : ISkype
- Sex : IProfile, IUser
- SilentMode : ISkype
- Smss : ISkype
- SpeedDial : IUser
- StartTime : IFileTransfer
- Status : IChat, ISmsTarget, IVoicemail, IFileTransfer, ISmsMessage, IChatMessage, ICall
- Stream : ICallChannel
- Streams : IApplication
- Subject : ICall
- t -

- TargetIdentity : ICall
- TargetNumbers : ISmsMessage
- Targets : ISmsMessage
- Text : ISmsChunk, ICallChannelMessage
- Timeout : ICommand, ISkype
- Timestamp : ICall, IChatMessage, IChat, IVoicemail, ISmsMessage
- Timezone : IUser, IProfile
- Topic : IChat
- TopicXML : IChat
- TransferActive : ICall
- TransferredBy : ICall
- TransferredTo : ICall
- TransferStatus : ICall
- Type : IVoicemail, IChat, ICallChannel, IGroup, ICall, IFileTransfer, IChatMessage, ISmsMessage
- u -

- User : ISkype
- Users : IChatMessage, IGroup
- UsersWaitingAuthorization : ISkype

Copyright © 2006 Skype Limited. All rights reserved.
- v -

- ValidatedSmsNumbers : IProfile
- Variable : ISkype
- Version : ISkype
- VideoIn : ISettings
- VideoReceiveStatus : ICall
- VideoSendStatus : ICall
- VideoStatus : ICall
- VmAllowedDuration : ICall
- VmDuration : ICall
- Voicemail : ISkype
- Voicemails : ISkype

Copyright © 2006 Skype Limited. All rights reserved.
- w -

- Wallpaper : IClient

Copyright � 2006 Skype Limited. All rights reserved.