Microsoft Chat Control Visual Basic Overview

Introduction

The Microsoft® Chat control, which is an ActiveX™-based control, allows the user to participate in a chat, or conversation, that can communicate both text and data between two or more users connected to a chat server. The Chat control provides a default user interface that is completely customizable, or a custom user interface can be designed using the Chat control to implement it.

Microsoft Visual Basic® Scripting Edition (VBScript) can be used to insert the Chat control into the HTML that creates a site on the World Wide Web (WWW or Web), or the Chat control can be used in a Microsoft Visual Basic application program to create a chat client. Both implementations enable conversations between multiple users on client computers accessing a common chat server over the Internet.

Software Requirements

To implement Web pages that use the Chat control, the Web server must have access to a chat server. The Chat control enables the connection from the user's computer to a chat channel, or chat room, on the chat server. For information on installing and configuring a chat server, see the *Internet Chat Server Operations Guide*.

User Interface

The Chat control allows a variety of user interface options. The control has a user interface that can be configured, or a custom user interface can be created using the control's automation interface to implement it.

The user interface provided with the Chat control is very flexible and can contain up to three panes in a single window. It includes a History text box, a Participant list box, a Send text box that provides space to type outgoing messages, a Send button, and a Whisper button. The Chat control can be set to allow users to resize the internal user interface (if it is being used).

To broadcast a message, the user clicks in the Send text box, types the text of the message, and clicks the Send button on the toolbar. To whisper to specific members of the chat room, a user can select the names of the members in the Participant list box, click in the Send text box, type the message, and click the Whisper button.

Installation

The Chat control can be installed from the Web or can be included with a Visual Basic application program.

Installation from the Web

Before users can take part in a conversation implemented with the Chat control, the control and certain supporting files must reside on the users' local computers.

When you include the control in a Web page, you can use the HTML OBJECT tag to download the control and the dynamic-link libraries (DLLs) that it depends on.

For example, the following lines inserted in a Web page cause the Chat control to be downloaded. Please note that the version in the **CODEBASE=** line is subject to change and should be updated to the current version of the control that is available.

```
<OBJECT
STANDBY="Downloading the Microsoft Chat control"
CODETYPE="application/x-oleobject"
CLASSID="clsid:D6526FE0-E651-11CF-99CB-00C04FD64497"
CODEBASE="MSChatOCX.Cab#Version=4,71,206,0"
WIDTH=600
HEIGHT=350
ID=Chat>
<PARAM NAME="UIOption"   VALUE="4095">
<PARAM NAME="Appearance" VALUE="0">
<PARAM NAME="BorderStyle" VALUE="0">
<PARAM NAME="BackColor"  VALUE="255">
</OBJECT>
```

Installation with a Visual Basic or Other Program

If the Chat control is used in a Visual Basic program or another OLE container, the Microsoft Visual Basic Setup Wizard can be used to create a setup program to install the control and the DLLs it requires.

Chat Control Rules

The following rules apply to the use of the Chat control:

- The user will not get the **OnMessage** event, if the message was sent by that user. However, if the internal user interface is used, the user will get the **OnTextMessageSent** event.
- If a critical error occurs while chatting, the user will automatically be ejected from the chat room.
- The **BorderStyle** and **Appearance** properties should be set only at startup time. They should not be changed in the middle of a conversation, because the content of the controls (text boxes and list boxes) would be lost.
- The user cannot ignore herself or himself or a host.
- Only a host can change somebody into a spectator, participant, or host.
- The host cannot change an ignored member into a host (the Ignore status must be removed first).
- The user cannot change the no whisper status of another user.
- The user cannot change herself or himself into a spectator, participant, or host.

Microsoft Chat Control Visual Basic Reference

Introduction

The Microsoft® Chat control is an ActiveX™-based control that provides the capability to create a chat, or conversation, that communicates text and/or data between two or more users connected through the Internet or an intranet.

The following document is a reference that contains the properties, methods, events, and error codes associated with using the Chat control in a Microsoft Visual Basic® or Microsoft Visual Basic Scripting Edition (VBScript) application.

Properties

This section covers the properties associated with the Microsoft Chat control.

Appearance Property

Returns or sets the appearance of the Chat control. This property is persistent and is read-only when connected to a chat room. The **Appearance** property has meaning only if the internal user interface is displayed.

object.**Appearance**[=*value*]

Parts

object
Required. An object expression that evaluates to a Chat control.

value
Optional. A short integer expression that determines the appearance of the Chat control. Can be set to one of the return values listed below.

Return Values

Returns a short integer value that indicates the current appearance of the Chat control. Can be one of the following values.

Value	Description
0	All flat.
1	3-D children only.
2	3-D control frame only.
3	All 3-D.

BackColor Property

Returns or sets the color of the background behind the text panes and the toolbar. This property is persistent and has meaning only if the internal user interface is being displayed.

object.**BackColor**[=*value*]

Parts

object
Required. An object expression that evaluates to a Chat control.

value
Optional. An OLE_COLOR expression that determines the background color. The default value is the system's default window background color.

Return Values

Returns the OLE_COLOR value of the current background behind the text panes and toolbar.

BorderStyle Property

Returns or sets the appearance of the border. This property is read-only when connected to a chat room and has meaning only if the internal user interface is displayed. This property is persistent.

object.**BorderStyle**[=*value*]

Parts

object
Required. An object expression that evaluates to a Chat control.

value
Optional. A short integer expression that determines the border appearance. Can be set to one of the return values listed below.

Return Values

Returns the current border style. Can be one of the following values.

Value	Description
0	None.
1	Single fixed for children only.
2	Single fixed for control frame only.
3	All single fixed.

Height Property

Returns or sets the height of the control in HiMetrics. Each HiMetrics unit corresponds to 0.01 mm. This property is persistent.

object.**Height**[=*value*]

Parts

object
Required. An object expression that evaluates to a Chat control.

value
Optional. A HiMetrics value that indicates the height of the Chat control.

Return Values

Returns the current height of the Chat control in HiMetrics units.

History Property

Returns the current content of the History text box. This property is read-only.

object.**History**

Parts

object
Required. An object expression that evaluates to a Chat control.

Return Values

Returns a BSTR value that contains the current contents of the History text box. If **UIOption** does not include the History text box (*value*=8), this property returns an empty string.

Remarks

The length of this string is limited by the **MaxHistoryLength** property.

LastMessageReceived Property

Returns the last message the user received from the channel. This property is read-only.

object.**LastMessageReceived**

Parts

object
Required. An object expression that evaluates to a Chat control.

Return Values Returns a BSTR value that contains the last message received by the user.

Remarks This property is updated whenever the user receives a text message.

LastMessageSent Property

Returns the last message sent by the user. This property is read-only.

object.**LastMessageSent**

Parts

object
Required. An object expression that evaluates to a Chat control.

Return Values Returns a BSTR value that contains the last text message sent by the user.

Remarks **LastMessageSent** is updated whenever the user sends a text message.

MaxHistoryLength Property

Returns or sets the maximum number of characters allowed in the History text box.

object.**MaxHistoryLength**[=*value*]

Parts

object
Required. An object expression that evaluates to a Chat control.

value
Optional. A long integer value that sets the maximum number of characters allowed in the History text box.

Return Values Returns the maximum number of characters allowed in the History text box. If **UIOption** does not include the History text box (*value*=8), this property is equal to zero. The default value of **MaxHistoryLength** is 32K.

MaxMembers Property

Returns or sets the maximum number of members in the chat room.

object.**MaxMembers**[=*value*]

Parts

object
Required. An object expression that evaluates to a Chat control.

value
Optional. A long integer expression that determines the maximum number of members allowed in the room. Only a host of the chat room can change this property.

Return Values Returns zero if the limit is the server's default limit; otherwise, returns the maximum number of members.

MaxMessageLength Property

Returns or sets the maximum number of characters allowed in the Send text box.

object.**MaxMessageLength**[=*value*]

Parts

object
Required. An object expression that evaluates to a Chat control.

value
Optional. A long integer value that determines the maximum number of characters allowed in the Send text box. The default value of **MaxMessageLength** is 2040 characters, which is the maximum message length on an MIC server.

Return Values Returns the maximum number of characters allowed in the Send text box. If **UIOption** does not include the Send text box (*value*=16), this property is equal to zero.

MemberCount Property

Returns the number of members currently in the user's chat room. This property is read-only.

object.**MemberCount**

Parts

object
Required. An object expression that evaluates to a Chat control.

Return Values

Returns zero if the user is not in a chat room; otherwise, it returns the number of members in the chat room.

Rating Property

Returns the rating of a chat room. This property is read-only.

object.**Rating**

Parts

object
Required. An object expression that evaluates to a Chat control.

Return Values

Returns a BSTR value that contains the Platform for Internet Content Selection (PICS) rating if the chat room is on an MIC server. Otherwise, this property returns an empty string. For more information on PICS ratings, see the *Internet Ratings* documentation.

RoomPath Property

Returns the location of the chat room to which the user is currently connected. This property is read-only.

object.**RoomPath**

Parts

object
Required. An object expression that evaluates to a Chat control.

Return Values

Returns a BSTR expression that evaluates to the server location of the chat room. The format of the server location is:

prefix://*server-name*[:*port-number*]/[[#]|[&]]*room-name*

The *prefix* can be mic or irc. The *server-name* is the host name of the chat server computer. The optional number sign (#) indicates that the chat room can be a global IRC room. The optional ampersand (&) indicates that the room can be a local IRC or MIC room. The *room-name* is the name of the chat room. If the user is not connected to a chat room, an empty string is returned.

RoomTopic Property

Returns or sets the chat room's topic. This property can be used to set the topic when creating a chat room, to change the topic when the user is in the room, or to get the current topic of the room that the user is in.

object.**RoomTopic**[=*value*]

Parts

object
Required. An object expression that evaluates to a Chat control.

value
Optional. A BSTR value that indicates the topic name to be set.

Return Values

Returns a BSTR value that contains the topic name of the chat room that the user is currently in.

Remarks

The topic cannot be set if the user is not the host of the chat room and the chat room was created with the value in the **EnterRoom** method set to (*Type*=2), which allows only the host to change the topic.

State Property

Returns whether the user is currently disconnected from, connected to, or in the process of connecting to a chat room. This property is read-only.

object.**State**

Parts

object
Required. An object expression that evaluates to a Chat control.

Return Values

Returns the current user state. Can be one of the following values.

Value	Description
1	Disconnected from any chat rooms.
2	Connecting to a chat room.
3	Connected to a chat room.

ThisParticipantAlias Property

Returns the user's alias. This property is read-only.

object.**ThisParticipantAlias**

Parts

object
Required. An object expression that evaluates to a Chat control.

Return Values

Returns a BSTR value that contains the user's nickname that was used when the **EnterRoom** method was called, if the user is currently in a chat room. If the user is not in a chat room, an empty string is returned.

ThisParticipantID Property

Returns the user's numeric identifier. This property is read-only.

object.**ThisParticipantID**

Parts

object
Required. An object expression that evaluates to a Chat control.

Return Values

Returns a long integer value that contains the user's numeric identifier, if the user is currently in a chat room. If the user is not in a chat room, the value is zero.

ThisParticipantName Property

Returns the user's Windows® user name. This property is read-only.

object.**ThisParticipantName**

Parts

object
Required. An object expression that evaluates to a Chat control.

Return Values

Returns a BSTR value that contains the user's Windows user name, if the user is currently in a chat room. If the user is not in a chat room, an empty string is returned.

UIOption Property

This property is a bitmask that contains flags that control the appearance of the user interface.

object.**UIOption**[=*value*]

Parts

object
Required. An object expression that evaluates to a Chat control.

value
Optional. A short integer expression that equals the sum of the desired user interface option values. Can include the following option values.

Value	Description
1	Set this value to display the Send button. Must set the Send text box (*value*=16) in order to set the Send button.
2	Set this value to display the Whisper button. Must set the Send button (*value*=1) in order to set the Whisper button.
4	Set this value to display the Participant list box.
8	Set this value to display the History text box. This window displays all messages that have been received.
16	Set this value to display the Send text box. This text box is where the user composes messages before sending them.
32	Set this value to allow members to be notified in the History text box when a new user joins the chat.
64	Set this value to allow members to be notified in the History text box when a user leaves the chat.
128	Set this value to allow the Chat control to display a dialog box that notifies the user

	when a member's status (host, participant, or spectator) changes, when a member's alias (nickname) changes, and when the user has been kicked by a host.
256	Set this value to insert a blank line between messages in the History text box.
512	Set this value to allow the Chat control to display error dialog boxes. When an **OnError** event is triggered, a dialog box containing the error message will appear.
1024	Set this value to allow the user to resize the control.
2048	Set this value to allow any URLs appearing in the History text box to be clickable.

Return Values Returns the sum of the option values currently selected.

Remarks To set a Chat control's user interface so that it includes a Send text box, Send button, Whisper button, History text box, and Participant list box, the **UIOption** property must be set to the sum of the option values. In this case, the sum would equal 31 (16 + 1 + 2 + 8 + 4 = 31). To set the control to have all options, the sum would equal 4095 (1 + 2 + 4 + 8 + 16 + 32 + 64 + 128 + 256 + 512 + 1024 + 2048 = 4095).

Width Property

Returns or sets the width of the control in HiMetrics. Each HiMetrics unit corresponds to 0.01 mm. This property is persistent.

object.**Width**[=*value*]

Parts

object
Required. An object expression that evaluates to a Chat control.

value
Optional. A HiMetrics value that indicates the width of the Chat control.

Return Values Returns the current width of the Chat control in HiMetrics units.

Methods

This section covers the methods associated with the Microsoft Chat control.

AboutBox Method

Displays the About message dialog box.

object.**AboutBox**

Parts

object
Required. An object expression that evaluates to a Chat control.

BanParticipant Method

Bans a member from or reinstates a member to a chat room.

object.**BanParticipant** *Name*, *Set*

Parts

object
Required. An object expression that evaluates to a Chat control.

Name
Required. A BSTR value that contains the alias of the member to be banned or reinstated.

Set
Required. A Boolean value that determines whether the member is banned or reinstated. If the value is set to TRUE, the member is banned from the chat room and cannot rejoin it. If the value is set to FALSE, the member is reinstated and can join the chat at any time.

Remarks

The **BanParticipant** method only prevents the member from rejoining or allows the member to rejoin the chat room after leaving or being kicked. This method does not kick the member from the chat room. **BanParticipant** cannot be called before the **OnEndEnumeration** event has been triggered.

CancelEntering Method

Cancels an **EnterRoom** operation

object.**CancelEntering**

Parts

object
Required. An object expression that evaluates to a Chat control.

Remarks

This method should be called only when the control's **State** property is equal to 2 (Connecting).

ClearHistory Method

Empties the History text box.

object.**ClearHistory**

Parts

object
Required. An object expression that evaluates to a Chat control.

Remarks

This method has an effect only if the value of **UIOption** includes a History text box (*value*=8).

EnterRoom Method

Allows the user to enter the specified chat room.

object.**EnterRoom** *RoomPath*, *RoomPwd*, *ThisParticipantAlias*, *SecurityPackage*, *Flags*, *Type*

Parts

object
Required. An object expression that evaluates to a Chat control.

RoomPath
Required. A string value containing the location of the chat room. The format of the string should be:

[*prefix:*]//*server-name*[:*port-number*]/[[#]|[&]]*room-name*

The *prefix* is optional and can be either mic or irc. The *server-name* is the host name of the chat server computer. The optional number sign (#) indicates that the chat room can be a global IRC room. The optional ampersand (&) indicates that the room can be a local IRC or MIC room. The *room-name* is the name of the chat room.

RoomPwd
Required. A string containing the room password or the host key used when creating or joining a password-protected room.

ThisParticipantAlias

Required. A string containing the alias that the user will be known by in the chat room.

SecurityPackage

Required. A string containing a list of security packages that are separated by semicolons and ordered by preference. If *SecurityPackage* is set to an empty string, the control asks the server for a list of security packages it provides. If the server replies no packages at all, an anonymous connection is tried. If the server replies with a list of packages, the first package on the list is tried. If the string is set to "ALL", all the server's security packages are tried in an order defined by the server. The control will stop looping through the security packages as soon as one package succeeds or an error not related to the account information occurs. If all packages fail, the control will try to connect the user anonymously. The "ANON" string can be used to request anonymous connections.

Flags

Required. A combination of values used when creating or joining a room. These values characterize the room type and cannot change dynamically once the room is created. Set to zero to use no flags. Can be any combination of the following values.

Value	Description
1	Join an existing chat room or create a chat room if one does not exist. If this flag is not set, the **EnterRoom** method will create a chat, but will not join an existing chat.
2	The chat room will not be exposed to other servers on the Internet.
4	The chat room is set up as read-only for news feeds, stock quotes, and so on.
8	The chat room will allow only MIC clients to join it. IRC users will not be able to join the chat room.
16	The server will accept only authenticated members in this chat room.
32	Only authenticated members can talk in the chat room.

Type

Required. A combination of values to set the type of chat room. Set to zero to use no flags. The chat room type can dynamically vary during its lifetime. The user is notified when the room type changes via the **OnRoomTypeChanged** event. Can be any combination of the following values.

Value	Description
1	Allows any user into the chat room.
2	Allows only hosts to change the topic of the

	chat room.
4	Allows only users who logged on to this server to enter the chat room.
8	Disables whispering in the chat room.
16	Creates an auditorium room. The member list shows only the user and the hosts. Participants cannot whisper, but hosts can whisper to anybody, regardless of whether the no whisper (*Type*=8) flag is set. Messages coming from participants are sent only to the hosts.
32	Creates a moderated room. Members join the chat room as spectators by default.

Remarks This method should be called only when the control's **State** property is equal to 1 (Disconnected).

ExitRoom Method

Disconnects the user from the current chat room.

object.**ExitRoom**

Parts

object
Required. An object expression that evaluates to a Chat control.

Remarks This method should be called only when the control's **State** property is equal to 3 (Connected).

GetParticipantRealName Method

Gets the real name of a participant by using the member's participant ID or alias (nickname).

object.**GetParticipantRealName** *ParticipantID*, *Alias*, *Synchronous*, *RealName*

Parts

object
Required. An object expression that evaluates to a Chat control.

ParticipantID
Required. A long integer value that indicates which member's real name is being requested. If *ParticipantID* is set to -1, the method uses the participant ID of the member selected in the Participant list box. If *ParticipantID* is not set to -1, the participant ID must have come from an **OnEnterParticipant** event. This value must be set to zero if the application is requesting the real name by using the member's alias.

Alias
Required. A BSTR value that contains the requested member's alias. This value must be set to an empty string if the application is requesting the real name by using the member's *ParticipantID*.

Synchronous
Required. A Boolean value that indicates whether this method is executed synchronously or asynchronously. If *Synchronous* is set to TRUE, the real name is put into the string that is pointed to by *RealName* for a Visual Basic or C/C++ application. If *Synchronous* is set to FALSE, the answer to the query comes back asynchronously. If the query is successful, the **OnParticipantRealName** event returns the information. If the query is not successful, the **OnError** event is triggered. VBScript does not support pointers, so *Synchronous* must be set to FALSE.

RealName
Required. An address of a string where the real name of the member will be stored if the method is executed synchronously.

Remarks
When using *ParticipantID*, the scope of the search is limited to members currently in the chat room. When using *Alias*, the entire server is searched.

InviteParticipant Method

Invites a member to the user's current chat room.

object.**InviteParticipant** *Alias*

Parts

object
Required. An object expression that evaluates to a Chat control.

Alias
Required. A string value that contains the alias of the member to invite.

Remarks
The member must be on the same chat server as the user.

KickParticipant Method

Kicks a member from the chat room.

object.**KickParticipant** *ParticipantID*, *KickComment*

Parts

object
Required. An object expression that evaluates to a Chat control.

ParticipantID
Required. A long integer value that indicates which member will be kicked. If *ParticipantID* is set to -1, the method uses the participant ID of the member selected in the Participant list box. If *ParticipantID* is not set to -1, the participant ID must have come from an **OnEnterParticipant** event.

KickComment
Required. A BSTR value that contains a text message with the host's reason for kicking the member. If the kicked member uses the Chat control and has the value 128 (inform about member status changes) set in the **UIOption** property, the member will see a message box with this reason.

Remarks

This method cannot be called before the **OnEndEnumeration** event has been triggered.

MoveSpitBar Method

Changes the appearance of the cursor to a cross hair so the user can resize the control by using the arrow keys or the mouse.

object.**MoveSpitBar**

Parts

object
Required. An object expression that evaluates to a Chat control.

SelectParticipants Method

Selects, deselects, or inverts the selection of members in the Participant list box.

object.**SelectParticipants** *value*

Parts

object
Required. An object expression that evaluates to a Chat control.

value
Required. A short integer value that determines which members are selected. Can be set to one of the following values.

Value	Description
0	Deselects any highlighted member.
1	Selects all the members.
2	Deselects the highlighted members and selects the members that were not highlighted.

Remarks

This method has an effect only if **UIOption** includes a Participant list box (*value*=4).

SendMessage Method

Sends a message.

object.**SendMessage** *DestinationList*, *Count*, *Message*

Parts

object
Required. An object expression that evaluates to a Chat control.

DestinationList
Required. A VARIANT value that indicates how this message should be distributed. The message can be broadcast to all members, whispered to one member, or whispered to a group of members:

- To broadcast the message to all members, the value must be set by an integer, a long integer, or an array of long integers (limited to one element) equal to -1.
- To whisper to one member (unicast), the value must be set by a long integer equal to the member's participant ID.
- To whisper to a group of members (multicast), the value must be set by an array of long integers corresponding to the member's participant ID.

Count
Required. A short integer that determines the number of recipients that will receive the message. To broadcast a message to all members, set the value of *Count* to zero. Otherwise, set *Count* to the number of members that will receive the message.

Message

Required. A VARIANT value that contains the message being distributed:

- To distribute a text message to be displayed in the History text box of the selected members, the data type must be a BSTR value that contains the text to be displayed.
- To distribute a data message being sent to the selected members, the data can be an array of bytes, shorts, longs, floats, doubles, Booleans, errors, currencies, or dates that correspond to the following VARIANT types: VT_UI1, VT_I2, VT_I4, VT_R4, VT_R8, VT_BOOL, VT_ERROR, VT_CY, VT_DATE.

Remarks

This method cannot be called before the **OnEndEnumeration** event has been triggered.

SetParticipantStatus Method

Changes the status of the member in a chat room.

object.**SetParticipantStatus** *ParticipantID*, *Mask*, *Status*

Parts

object

Required. An object expression that evaluates to a Chat control.

ParticipantID

Required. If *ParticipantID*=-1, the member selected in the Participant list box will be affected. Only one member can be selected for this method to work. Otherwise, the value of *ParticipantID* has to come from an **OnEnterParticipant** event.

Mask

Required. A short integer indicating the types of status changes that the user wants to make. Set to zero if no masks are wanted. Can be set to any combination of the following values.

Value	Description
1	Must be set if *Status* is 1, 2, or 4.
2	Must be set if *Status* is changed to 8 or zero (which would allow whispers).
4	Must be set if *Status* is changed to 16 or zero (which would allow the user to receive messages from the member again).

Status

Required. A short integer setting the participant's status changes. Set to zero if the user wants to allow the member to whisper to the user (if *Mask* includes 2) or

allow the user to receive messages from the member (if *Mask* includes 4). Can be set to any combination of the following values.

Value	Description
1	Changes the status of the member to a host.
2	Changes the status of the member to a participant.
4	Changes the status of the member to a spectator.
8	Changes the status of the member to not allow whispers.
16	Changes the status of the member to Ignored, so the user will not receive any messages from that member.

Remarks

This method cannot be called before the **OnEndEnumeration** event. The following are the rules that apply to status changes:

- The user cannot ignore herself or himself or the hosts.
- The user can set the flag for no whispers (*Status*=8) only on herself or himself.
- Only a host can change somebody's privileges.

Events

This section covers the events associated with the Microsoft Chat control.

OnBeginEnumeration Event

Occurs when entering a chat room, before the list of members already in the conversion appears in the Participant list box.

*object*_**OnBeginEnumeration**

Values

object
Required. An object expression that evaluates to a Chat control.

OnEndEnumeration Event

Occurs after getting the initial list of members in the conversation, including the user.

*object*_**OnEndEnumeration**

Values

object
Required. An object expression that evaluates to a Chat control.

Remarks

The user can start to use the **ThisParticipantID** property or the **SetParticipantStatus**, **KickParticipant**, **BanParticipant**, and **SendMessage** methods once this event is triggered.

OnEnterParticipant Event

Occurs when either the list of members already in the conversation first appears when entering a chat room or when a new participant joins the conversation.

*object*_**OnEnterParticipant(***ParticipantID***,** *Name***,** *Status***)**

Values

object
Required. An object expression that evaluates to a Chat control.

ParticipantID
A long integer value that contains the participant ID of the member entering the chat room.

Name
A BSTR value that contains the alias of the member joining the chat room.

Status
A short integer value indicating the status of the member entering the room. Can be one of the following values.

Value	Description
1	Member is a host.
2	Member is a participant.
4	Member is a spectator.

OnError Event

Occurs when there is an error in the control. If the display dialog boxes flag (*value*=512) is set in **UIOption**, the error message is displayed to the user.

*object*_**OnError(***ErrorCode***,** *Description***)**

Values

object
Required. An object expression that evaluates to a Chat control.

ErrorCode
A short integer value that contains the numeric value of the error code.

Description
A BSTR value that contains the error message that is associated with the error code.

OnExitParticipant Event

Occurs when a member leaves the chat room.

*object*_**OnExitParticipant(***ParticipantID***)**

Values

object
Required. An object expression that evaluates to a Chat control.

ParticipantID
A long integer value that contains the participant ID of the member exiting the chat room.

OnHistoryFull Event

Occurs each time a message is added to the History text box, when the total number of characters in the History text box is at least 90 percent of the **MaxHistoryLength** property (which is the maximum number of characters allowed in the History text box).

*object*_**OnHistoryFull(***Percent***)**

Values

object
Required. An object expression that evaluates to a Chat control.

Percent
The exact percentage of the History text box's contents that have been used.

Remarks

If the History text box gets too full to contain an incoming message, the top 20 percent of the text box's contents are removed. If this does not give enough space to fit the incoming message, more text is removed.

OnMessage Event

Occurs when a user gets a message from a member or the chat room.

*object*_**OnMessage(***SenderID***,** *Message***,** *MessageType***)**

Values

object
Required. An object expression that evaluates to a Chat control.

SenderID
A long integer that contains the participant ID of the member who sent the message, or zero if the message comes from the chat room.

Message
A VARIANT value that contains the message. The incoming message can be a text message (BSTR) or data (an array of bytes, shorts, longs, floats, doubles, Booleans, errors, currencies, or dates).

MessageType
Can be a combination of the following values.

Value	Description
0	A text or data message that is broadcast to all members.
1	A text or data message that is whispered to one or more members.
2	A ComicChat thought, which is text only.
4	An action, which is text only.

OnParticipantAliasChanged Event

Occurs when the member's alias is changed. Also, if the display notifications flag (*value*=128) is set in the **UIOption** property, a dialog box is displayed that says "<OldAlias> is now known as <NewAlias>."

*object*_**OnParticipantAliasChanged(***ParticipantID***,** *OldAlias***,** *NewAlias***)**

Values

object
Required. An object expression that evaluates to a Chat control.

ParticipantID
A long integer value that contains the participant ID of the member whose alias is changing.

OldAlias
A BSTR value that contains the old alias.

NewAlias
A BSTR value that contains the new alias.

OnParticipantInvited Event

Occurs when a member invites the user to the member's chat room.

*object*_**OnParticipantInvited(***RoomName***,** *Alias***)**

Values

object
Required. An object expression that evaluates to a Chat control.

RoomName
A string value containing the name of the member's chat room.

Alias
A string value containing the name of the member who invited the user to the member's chat room.

OnParticipantKicked Event

Occurs when the user is kicked from the chat room.

*object*_**OnParticipantKicked(***ParticipantID***,** *Reason***)**

Values

object
Required. An object expression that evaluates to a Chat control.

ParticipantID
A long integer value that contains the participant ID of the host who kicked the user from the chat room.

Reason
A BSTR value that contains the reason given for kicking the user from the chat room. This value can be an empty string, if no reason was given.

OnParticipantRealName Event

Occurs when you make a successful call to **GetParticipantRealName** with the *Synchronous* parameter set to FALSE.

*object*_**OnParticipantRealName(***Alias***,** *RealName***)**

Values

object
Required. An object expression that evaluates to a Chat control.

Alias
A BSTR value that contains the current alias of the member.

RealName
A BSTR value that contains the real name of the member.

OnParticipantStatusChanged Event

Occurs when a member changes status.

*object*_**OnParticipantStatusChanged(***ParticipantID***,** *Status***)**

Values

object
Required. An object expression that evaluates to a Chat control.

ParticipantID
A long integer value that contains the participant ID of the member whose status is changing.

Status
A short integer value that indicates the new status of the member. The new status is a combination of the following values.

Value	Description
1	Member has become a host.
2	Member has become a participant.
4	Member has become a spectator.
8	Member does not allow whispers.
16	Member is ignored, so all incoming messages from this member are discarded.

Remarks

Any member is either a host (*Status*=1), participant (*Status*=2), or spectator (*Status*=4), so a valid status must include one of these values.

OnRoomTopicChanged Event

Occurs when the room topic property changes.

*object*_**OnRoomTopicChanged(***NewRoomTopic***)**

Values

object
Required. An object expression that evaluates to a Chat control.

NewRoomTopic
A BSTR value that contains the description of the new topic of conversation.

OnRoomTypeChanged Event

Gives the value of the new room type.

*object*_**OnRoomTypeChanged(***NewRoomType***)**

Values

object
Required. An object expression that evaluates to a Chat control.

NewRoomType
A short integer value that equals the sum of the room type settings. Can be a combination of the following values.

Value	Description
1	Allows any user into the chat room.
2	Allows only hosts to change the topic of the chat room.
4	Allows only users who logged on to this server to enter the chat room.
8	Disables whispering in the chat room.
16	Creates an auditorium room. The member list shows only the user and the hosts. Participants cannot whisper, but hosts can whisper to anybody, regardless of whether the no whisper (*NewRoomType*=8) flag is set. Messages coming from participants are sent only to the hosts.
32	Creates a moderated room. Members join the chat room as spectators by default.

Remarks

For example, if *NewRoomType*=10, it means that only hosts can change the room topic (*NewRoomType*=2), and no whispers are allowed in the room (*NewRoomType*=8).

OnStateChanged Event

Occurs when the user's state changes.

*object*_**OnStateChanged(***NewState***)**

Values

object
Required. An object expression that evaluates to a Chat control.

NewState
A short integer that indicates the user's new state. Can be one of the following values.

Value	Description
1	Disconnected from any chat rooms.
2	Connecting to a chat room.
3	Connected to a chat room.

OnTextMessageSent Event

Occurs whenever the user sends a text message by using the internal Send button or Whisper button.

*object*_**OnTextMessageSent(***TextMessage***,** *MessageType***)**

Values

object
Required. An object expression that evaluates to a Chat control.

TextMessage
A BSTR value that contains the text sent by the user.

MessageType
A short integer value that indicates the type of message sent. Can be one of the following values.

Value	Description
0	The user sent some text as a broadcast to all members.
1	The user sent some text as a whisper to one or more members.

Remarks

This event is NOT triggered when using the **SendMessage** method.

Error Codes

The following list contains the error codes and their associated error messages. These error messages are subject to change and may not be exactly as they appear below.

Error code	Error message
5	Illegal function call.
6	Overflow error.
7	Out of memory.
70	Permission is denied.
380	Invalid property value.
382	Property is read-only at run time.
383	Property is read-only.
394	Property is write-only.
1000	An error occurred. Please leave the application and try again.

1001	Could not load the RICHED32.DLL library.
1002	The room could not be found.
1003	You are already trying to enter a room.
1004	You are not trying to enter a room.
1005	You are already in a chat room.
1006	The room is full. Please try again later.
1007	You cannot create this room, it already exists.
1008	Could not make the room unique.
1009	Too many rooms are already open on this server.
1010	This is a room for which you need an invitation.
1011	This is a password protected room. The room password specified is invalid.
1012	The operation is denied.
1013	The chat room path specified is invalid.
1014	Can't connect to the chat server.
1015	User unknown. Please verify your entries.
1016	The alias specified is already in use.
1017	The name specified is too long.
1018	The user password specified is too long.
1019	The room password specified is too long.
1020	The alias specified is too long.
1021	The room name specified is too short. Please enter a longer name.
1022	The room name specified is too long. Please enter a shorter name.
1023	The server name specified is too long.
1024	The chat room topic specified is too long. Please enter a shorter topic.
1025	A string specified is too long.
1026	The *Flags* parameter specified is invalid.
1027	The *Type* parameter specified is invalid.
1028	The room name specified contains illegal characters.
1029	The alias specified contains illegal characters.
1030	For MIC and IRC local rooms, the first character has to be &. For IRC global rooms, the first character has to be #.
1031	This chat has a bad name. The server cannot create it.

1032	This user is unknown. The authentication failed.
1033	Select the name of the member you want to ignore, and try again. You cannot ignore yourself or a host.
1034	The participant ID specified is invalid.
1035	Please select exactly one member and try the operation again.
1036	You cannot kick yourself!
1037	You cannot ban or reinstate yourself!
1038	The status specified is invalid.
1039	The mask specified is invalid.
1040	The status change was denied.
1041	Only a host can complete this operation.
1042	You can only complete this operation on yourself.
1043	This operation cannot be attempted on a host.
1044	This operation cannot be attempted on an ignored member.
1045	You cannot whisper to yourself.
1046	Could not send the message.
1047	The Receive text box limitation is too small to display the message. You might want to increase this limitation.
1048	The destination list specified is invalid.
1049	The message type specified is invalid.
1050	The message format is invalid.
1051	You are trying to whisper to too many recipients.
1052	The message is empty.
1053	The message to send is too long and could not be sent.
1054	Only a system operator can complete this operation.
1055	You are currently not connected to a chat server.
1056	The chat server specified could not be found.
1057	You are currently not in a room.
1058	The connection to the chat server was lost.
1059	A host of this chat room closed the room. You are no longer in the room.
1060	Property is read-only when you're in a chat room.

1061	The authentication could not be performed.
1062	The password provided is invalid.
1063	You are not allowed to whisper in this chat room.
1064	This member does not accept whispers.
1065	All the recipients refuse whispers.
1066	The member name specified is blank or invalid.
1067	This server does not allow IRC clients.
1068	You have been banned from this chat room.
1069	This chat room only allows authenticated members.
1070	This server is full and does not allow additional members. Please try again later.
1071	The server does not recognize the security package specified.
1072	The *SecurityPackage* parameter specified is too long.
1073	An MIC operation was attempted on an IRC channel
1074	The PICS rating of this channel doesn't allow you to complete this operation.
1075	No member with the alias specified could be found.
1076	Couldn't get the real name of the member.
1077	An error occurred on the chat server.
1078	You cannot put carriage returns in a message sent to an IRC room.
1079	A security-related error occurred on the server.
1080	This MSN account is unknown.
1081	This NT account is unknown.
1082	The authenticated login failed.
1083	Could not log you in successfully.
1084	This chat program cannot join this type of room. Please try another room.
1085	The chat room could not be created.
1086	The operation could not be performed successfully.
1087	Only administrators can perform this operation.
1088	The server and client protocols in use are not compatible.
1089	A member is trying to flood you.

1090	You are not allowed to talk in this room.
1091	The member is already in this room.
1092	The member you specified is currently unknown on this chat server.

Copyrights & Trademarks