Metreos Communications Environment

Framework Package Library API Reference

August 2004

Version 1.1

TABLE OF CONTENTS

1. CALL CONTROL

Actions

Metreos Framework Package Library

Metreos.CallControl.AnswerCall Action

Answer an incoming 1st party phone call.

Operation

Asynchronous

Required Parameters

Parameter Name	Data Type	Description
answer	System.Boolean	A boolean value indicating whether to answer the call or not.
callId	System.String	A unique token used to identify this particular call leg in future actions and events.
userData	System.String	An opaque token used to correlate asynchronous actions with final responses in an application. Leave this at the default 'none' unless you need to change this.

Optional Parameters

None.

Result Data Fields

Parameter Name	Data Type	Description
callId	System.String	A unique token used to identify this particular call leg in future actions and events

Return Value

Returns "success" if the call is being answered; otherwise, "failure" is returned.

Remarks

This action answers an incoming call based on the value of the boolean parameter "answer". If answer is set to "true" the call will be answered normally. If answer is set to "false" the incoming call will be rejected. Once all negotiations have been completed with the caller an "AnswerCall_Complete" event will be fired to the application. If an error occurs and the call can not be answered an "AnswerCall_Failed" event will be fired to the application. If the application requires full-duplex media, invoke "SetMedia".

Requirements

Metreos Framework version 1.1

Metreos H.323 Provider version 1.1

Metreos.CallControl.Hangup Action

Terminate an existing 1st party phone call.

Operation

Asynchronous

Required Parameters

Parameter Name	Data Type	Description
callId	System.String	A unique token used to identify this particular call leg in future actions and events
userData	System.String	An opaque token used to correlate asynchronous actions with final responses in an application. Leave this at the default 'none' unless you need to change this.

Optional Parameters

None.

Result Data Fields

Parameter Name	Data Type	Description
callId	System.String	A unique token used to identify this particular call leg in future actions and events

Return Value

Returns "success" if the call is being terminated; otherwise, "failure" is returned.

Remarks

This action terminates the existing call with the specified call ID. This action should only be used with calls which have been successfully established and should not be used to reject incoming calls. To reject an incoming call, use "AnswerCall" with the answer parameter set to false.

Once the application server has finalized the hang up a "Hangup_Complete" event will be fired to the application. If an error occurs and the call could not be terminated a "Hangup_Failed" event will be fired to the application.

Requirements

Metreos Framework version 1.1

Metreos H.323 Provider version 1.1

Metreos.CallControl.MakeCall Action

Initiate a 1st party phone call from the application server to a remote party.

Operation

Asynchronous

Required Parameters

Parameter Name	Data Type	Description
mediaIP	System.String	The IP address of the media server that will be terminating media for this call
mediaPort	System.UInt16	The port number on the media server where the media for this call should be sent
to	System.String	The fully-qualified address of the remote party to be called
userData	System.String	An opaque token used to correlate asynchronous actions with final responses in an application. Leave this at the default 'none' unless you need to change this.

Optional Parameters

Parameter Name	Data Type	Description
from	System.String	A friendly name for the originator of this call (generally, this should be set to the application's name)
connectionId	System.Int32	Media server connection ID associated with this call.

Result Data Fields

Parameter Name	Data Type	Description
callId	System.String	A unique token used to identify this particular call leg in future actions and events

Return Value

Returns "success" if the call was initiated; otherwise, "failure" is returned.

Remarks

This action initiates a call from the application server to the location specified in the "to" parameter. It is assumed that appropriate media resources have been reserved on the media server prior to this action being performed. The media details should be passed in as the "mediaIp" and "mediaPort" parameters.

One the application server has fully established the call a "MakeCall_Complete" event will be fired to the application. If the call can not be established then a "MakeCall_Failed" event will be fired to the application.

Requirements

Metreos Framework version 1.1

Metreos H.323 Provider version 1.1

Metreos.CallControl.SetMedia Action

Set the media information to be used when answering an incoming 1st party phone call.

Operation

Synchronous

Required Parameters

Parameter Name	Data Type	Description
callId	System.String	A unique token used to identify this particular call leg in future actions and events
mediaIP	System.String	The IP address of the media server that will be terminating media for this call
mediaPort	System.UInt16	The port number on the media server where the media for this call should be sent

Optional Parameters

None.

Result Data Fields

Parameter Name	Data Type	Description
callId	System.String	A unique token used to identify this particular call leg in future actions and events

Return Value

Returns "success" if the information was set; otherwise, "failure" is returned.

Remarks

Use this action to inform the call control stack what IP address and port for RTP media to use when establishing an incoming call. This action should be called before each "AnswerCall". Usually, the media values will be the result of a media server half-connect action (See "CreateConnection" in the Media Server section). If the call is going to be rejected, set the "mediaIp" and "port" parameters to 0.

Due to technical details of the H.323 stack, this call is divorced from "AnswerCall". It is imperative that this action be called as quickly as possible after the "IncomingCall" event has been received since the provider will only wait on it for a finite amount of time. After that, there can be a reasonably longer delay before "AnswerCall" is executed. This gives time for complex logic to be performed to determine whether the call is to be answered or rejected without running into issues with the provider's "IncomingCall" timer.

Requirements

Metreos Framework version 1.1

Events

Metreos Framework Package Library

Metreos.CallControl.GotDigits Event

Event fired indicating that DTMF digits have been observed on the call signaling path.

Type

Non-Triggering

Event Parameters

Parameter Name	Data Type	Description
callId	System.String	A unique token used to identify this particular call leg in future actions and events
digits	System.String	A string of DTMF digits

Remarks

This event will fire for each digit received over the call control signaling channel. For these digits to affect media processing termination conditions those digits must be inserted into the media processing stream.

See “Metreos.MediaServer.SendDigits” in the “Media Control” section for more details.

Requirements

Metreos Framework version 1.1

Metreos H.323 Provider version 1.1

Metreos.CallControl.Hangup Event

Event fired indicating that an existing 1st party phone call has ended without the application's request.

Type

Non-Triggering

Event Parameters

Parameter Name	Data Type	Description
callId	System.String	A unique token used to identify this particular call leg in future actions and events
from	System.String	The directory number of the remote calling party
reason	System.String	The reason the call was terminated
to	System.String	The dialed number of the local party being called

Remarks

When this event is received, the call has been terminated. No further action is necessary. The "callId" parameter indicates which call leg has been terminated. The "reason", "to", and "from" parameters are intended only for convenience when displaying output to a human user.

Generally this event will fire in response to the remote party physically placing their terminal on hook; however, it is possible for this event to fire if the underlying call control stack terminates the call for network or other call signaling reasons. In this situation, the "reason" parameter will indicate why the call was terminated. It is returned in certain Metreos.CallControl events and has one of 5 values, each of which correlate to a condition which caused the event to occur. The events that return a "reason" parameter are MakeCall_Failure and Hangup.

Reason codes and list of possible conditions which could have caused each reason code:

"NormalCallClearing"

- Ended By Local User
- Ended By Remote User
- Ended By Caller Abort

"NoAnswer"

- Ended By No Answer
- Ended By Answer Denied
- Ended By Refusal

"RemoteBusy"

- Ended By Remote Busy

"Unreachable"

- Ended By Unreachable
- Ended By Local Congestion
- Ended By Host Offline
- Ended By No Bandwidth
- Ended By No User
- Ended By Remote Congestion

"OtherOrUnknown"

- All other cases

Requirements

Metreos Framework version 1.1

Metreos H.323 Provider version 1.1

Metreos.CallControl.IncomingCall Event

Event fired indicating that a remote party is attempting to establish a 1st party phone call with the application server.

Type

Triggering

Event Parameters

Parameter Name	Data Type	Description
callId	System.String	A unique token used to identify this particular call leg in future actions and events
from	System.String	The directory number of the remote calling party
to	System.String	The dialed number of the local party being called

Remarks

This event signals the beginning of a call. It provides the remote party's address as well as the local identity that party is trying to reach. Both "SetMedia" and "AnswerCall" actions should be invoked in response to this event.

If the call is intended to be answered, the normal case is to perform a "half-connect" operation on the media server to reserve the necessary media resources by executing "Metreos.Provider.MediaServer.CreateConnection" with the remote party's media information set to zeros. Then, use "SetMedia" to inform the call control provider of these resources. Next, "AnswerCall" should be executed with the "answer" parameter set to "true".

If the call is to be rejected, "SetMedia" should be executed with all media information set to zero. Then, "Answer" should be executed with the "answer" parameter set to false.

The "callId" parameter contains the unique token that should be used when performing any actions on this call leg. For instance, if the application wishes to terminate this call, then a "Hangup" action would be invoked with the "callId" from this event.

Requirements

Metreos Framework version 1.1

Metreos H.323 Provider version 1.1

Metreos.CallControl.SignalingChange Event

Event fired indicating that the remote party has re-negotiated their media parameters.

Type

Non-Triggering

Event Parameters

Parameter Name	Data Type	Description
callId	System.String	A unique token used to identify this particular call leg in future actions and events
mediaIP	System.String	The IP address of the remote party that will be terminating media for this call
mediaPort	System.Int32	The port number on the remote party where the media for this call should be sent

Remarks

This event will only occur after a call has been successfully established. It is typically fired in response to a call signaling message indicating that the remote party has placed the call on hold, taken the call off hold, or transferred the call to a different endpoint.

In response to this event the application would typically send the new media information to the media server so that the full-duplex media channel may be preserved. Execute a "Metreos.Providers.MediaServer.CreateConnection" action and specify the connection ID which was returned from the initial invocation of the action along with the new media parameters. See the "Media Control" section for more details.

Requirements

Metreos Framework version 1.1

Metreos H.323 Provider version 1.1

Asynchronous Callback Events

Metreos Framework Package Library

Metreos.CallControl.AnswerCall_Complete Event

Event fired indicating that the "AnswerCall" action completed successfully.

Type

Asynchronous Callback

Event Parameters

Parameter Name	Data Type	Description
callId	System.String	A unique token used to identify this particular call leg in future actions and events
from	System.String	The directory number of the remote calling party
to	System.String	The dialed number of the local party being called

Remarks

This event indicates that an inbound, 1st party phone call has been fully connected to the application server. Applications may perform further actions on this call leg by invoking call control actions with the "callId" parameter set to the value received in this event.

Requirements

Metreos Framework version 1.1

Metreos H.323 Provider version 1.1

Metreos.CallControl.AnswerCall_Failed Event

Event fired indicating that the "AnswerCall" action failed to complete.

Type

Asynchronous Callback

Event Parameters

Parameter Name	Data Type	Description
callId	System.String	A unique token used to identify this particular call leg in future actions and events

Remarks

The call control stack was unable to finish answering the call. No further action is required by the application. The application should no longer attempt call control actions on this call leg as the value of the "callId" parameter is no longer an active call.

Requirements

Metreos Framework version 1.1

Metreos H.323 Provider version 1.1

Metreos.CallControl.Hangup_Complete Event

Event fired indicating that the "Hangup" action completed successfully.

Type

Asynchronous Callback

Event Parameters

Parameter Name	Data Type	Description
callId	System.String	A unique token used to identify this particular call leg in future actions and events

Remarks

The hang up operation has completed successfully. Any state related to the call can now be safely cleaned up. The application should no longer attempt call control actions on this call leg as the value of the "callId" parameter is no longer an active call.

Requirements

Metreos Framework version 1.1

Metreos H.323 Provider version 1.1

Metreos.CallControl.Hangup_Failed Event

Event fired indicating that the "Hangup" action completed unsuccessfully.

Type

Asynchronous Callback

Event Parameters

Parameter Name	Data Type	Description
callId	System.String	A unique token used to identify this particular call leg in future actions and events

Remarks

The call control stack was unable to hang up the specified call leg. Upon receiving this event the application should no longer reference the call ID that was used in the original hang up operation and should consider the call to be terminated.

Requirements

Metreos Framework version 1.1

Metreos H.323 Provider version 1.1

Metreos.CallControl.MakeCall_Complete Event

Event fired indicating that the "MakeCall" action completed successfully.

Type

Asynchronous Callback

Event Parameters

Parameter Name	Data Type	Description
callId	System.String	A unique token used to identify this particular call leg in future actions and events
mediaIP	System.String	The IP address of the remote party that will be terminating media for this call
mediaPort	System.String	The port number on the remote party where the media for this call should be sent

Remarks

This event indicates successful connection with the remote party for an application server initiated, 1st party phone call. Normally, the application would initiate a "full connect" to the media server at this time by executing "Metreos.Providers.MediaServer.CreateConnection" with the remote party's media information filled in from the parameters of this event. Once that is done, media will be flowing bi-directionally between both devices.

Requirements

Metreos Framework version 1.1

Metreos H.323 Provider version 1.1

Metreos.CallControl.MakeCall_Failed Event

Event fired indicating that the "MakeCall" action failed to complete.

Type

Asynchronous Callback

Event Parameters

Parameter Name	Data Type	Description
callId	System.String	A unique token used to identify this particular call leg in future actions and events
from	System.String	The directory number of the remote calling party
reason	System.String	The reason the call was terminated.
to	System.String	The dialed number of the local party being called

Remarks

The "MakeCall" action has failed due to a rejection by the remote party or ring out conditions. No further action is required. The application should no longer attempt call control actions on this call leg as the value of the "callId" parameter is no longer an active call.

Refer to the description of the "Hangup" event for an explanation of and the possible values for the "reason" parameter.

Requirements

Metreos Framework version 1.1

Metreos H.323 Provider version 1.1

2. MEDIA CONTROL

The Media Server responds with a numeric result code after completing a request. The following table enumerates these codes:

Result Code	Reason	Description
0	Success	The action completed successfully
1	Provisional OK	The action is executing asynchronously
4	All sessions are in use	The maximum configured number of connections has been reached
5	Server not accepting commands	Media Server is in shutdown state
6	Internal error	Internal logic error
7	Device error	Media firmware error
8	Media resource not available	All conference or voice resources are busy
9	Internal error	Internal call state transition error
10	Internal error	Internal event table i/o error
11	Internal error	Internal asynchronous event state error
12	Session timeout	No activity on session within the configured timeout interval
13	Action timeout	An action did not complete within the configured timeout interval
14	Session busy	The connection is busy with another request
15	Already connected	A connection with the supplied ID already exists
16	No connection exists	A Media Server operation was requested for a connection ID which does not exist
20	Unrecognized event	Internal error: unexpected media termination event
21	Nonexistent command	An invalid command ID was specified
22	Connection ID invalid format	The specified connection ID was not well-formed, e.g. negative
23	Connection ID not registered	The specified ID does not match any current connection
24	Session is not in conference	A conference operation was requested on a connection which is not in conference
25	Reserved for future use	
26	Too few parameters supplied	Not all expected parameters were supplied
27	Value error	The value of an action parameter was outside the parameter's valid range of values
30	File open error	A specified voice file could not be opened
31	File read or write error	A specified voice file could not be read
35	Malformed request	A protocol block used to transport a request to the Media Server was malformed
36	Invalid termination condition	Unrecognized termination condition supplied

Actions

Metreos Framework Package Library

Metreos.Providers.MediaServer.CreateConnection Action

Create a voice-capable connection to the media server. Optionally, this action can be used to reserve a port on a media server for later use.

Operation

Synchronous

Required Parameters

Parameter Name	Data Type	Description
connectionId	System.Int32	Connection ID associated with this connection

Optional Parameters

Parameter Name	Data Type	Description
remoteIp	System.String	IP address of the remote media endpoint
remotePort	System.UInt16	Port on which the remote media endpoint is listening for media
connectionAttribute	System.String	Any additional conference attributes can be specified in this field
callId	System.String	Call ID associated with this connection

Result Data Fields

Parameter Name	Data Type	Description
connectionId	System.String	Unique Media Server ID for a connection
ipAddress	System.String	Local media IP address
port	System.String	Local media port
resultCode	System.String	Result of a Media Server operation

Return Value

Returns “success” if the connection was successfully created; otherwise “failure” is returned. If the media server fails to create the connection, the “resultCode” parameter will hold an integer value indicating the reason why the connection could not be created.

Remarks

Applications that intend to use the media capabilities exposed by the Metreos Communications Environment must first create a connection to the media server. Media server connections are specific to a distinct user of an application. For instance, if a conferencing application wishes to conference four individual phone calls then four distinct media server connections must be created first, and later put into the same conference.

Sometimes it is necessary for an application to reserve a media port before knowing the remote party's media details. Also known as a half-connection, this allows the application to be sure that a media port is available as well as retrieves the media server's media attributes for passing on to the remote party of the application. To execute a half-connection issue a "CreateConnection" action with "0" as the "connectionId", "remoteIp" and "remotePort" parameters. When the media server sees this it will reserve a port and return the IP address and port number that it is expecting media on for the new connection.

Half-connections allow the application developer to pass on the media server's information to the remote party, effectively setting up a half-duplex connection. To transition to a full-duplex connection the application was issue a second "CreateConnection" action, this time passing in the connection ID returned in the original "CreateConnection". Along with the connection ID the remote party's media details must also be passed into "remoteIp" and "remotePort". Once this is done, the media server will have the necessary information to transmit media to the remote party.

If the remote parties media parameters change and need to be reset on the media server, issue a "CreateConnection" action with the existing connection ID and the new media parameters specified in "remoteIp" and "remotePort".

Requirements

Metreos Framework version 1.1

Metreos Media Server Provider version 1.1

Metreos.Providers.MediaServer.CreateConnectionConference Action

Create a media server connection and immediately place that connection into a conference. Optionally, if the conference does not exist, create a new one.

Operation

Synchronous

Required Parameters

None.

Optional Parameters

Parameter Name	Data Type	Description
conferenceId	System.Int32	Unique Media Server ID for a conference
connectionId	System.Int32	Unique Media Server ID for a connection
remoteIp	System.String	IP address of the remote media endpoint
remotePort	System.UInt16	Port on which the remote media endpoint is listening for media
soundToneOnJoin	System.Boolean	Indicates whether tones are played when participants join the conference

Result Data Fields

Parameter Name	Data Type	Description
conferenceId	System.String	conferenceId associated with this action
connectionId	System.String	Unique Media Server ID for a connection
ipAddress	System.String	Media Server IP address
port	System.String	Media Server port
resultCode	System.String	Result of a Media Server operation

Return Value

Returns “success” if the connection was created and added to a conference; otherwise, “failure” is returned. If the media server fails to create the connection, the “resultCode” parameter will hold an integer value indicating the reason why the connection could not be created.

Remarks

This action allows the application to create a connection to the media server and at the same time either establish a new conference or add the newly created connection into an existing conference.

This action should not be used to perform a “half-connect” to the media server. If that is necessary use “CreateConnection” and once the media information for the remote party is known use “CreateConnectionConference”.

When creating a new conference, specify “0” for conferenceId to obtain a

conferenceId for the newly created connection in the returned conferenceId field. If you specify “0” for conferenceId, and provide a valid non-zero connectionId, a new conference will be created, and the connection with the specified connectionId will be placed in the conference. If both conferenceId and connectionId are specified, the connection associated with connectionId will be placed in the conference associated with conferenceId. If you’re providing a connectionId, you may also specify remoteIp and remotePort that the remote endpoint is expecting media on. In other words, you can complete a half-connection by providing the connectionId, remoteIp and remotePort to CreateConnectionConference

The returned connectionId is merely a copy of the one that we passed in. The returned ipAddress and port are the ipAddress and port

Requirements

Metreos Framework version 1.1

Metreos Media Server Provider version 1.1

Metreos.Providers.MediaServer.DeleteConnection Action

Deletes an existing media server connection freeing the resources used by that connection for use by other applications.

Operation

Synchronous

Required Parameters

Parameter Name	Data Type	Description
conferenceId	System.Int32	Unique Media Server ID for a conference
connectionId	System.Int32	Unique Media Server ID for a connection

Optional Parameters

None

Result Data Fields

Parameter Name	Data Type	Description
resultCode	System.String	Result of a Media Server operation

Return Value

Returns "success" if the connection was removed; otherwise, "failure" is returned. If the media server fails to delete the connection, the "resultCode" parameter will hold an integer value indicating the reason why the connection could not be deleted.

Remarks

Once an application is finished using a media server connection it must delete that connection to free all resources that were being used by the connection. If an application creates more than one connection then all of the connections that it created must be properly deleted.

Applications may delete connections at any time, regardless of whether an operation such as playing an announcement is currently executing on that media server connection. If an application does delete a connection that is currently executing an asynchronous media server operation then the application will not receive a final asynchronous event for that operation. For example, if an announcement is being played to a connection and the application deletes that connection, no "PlayAnnouncement_Complete" will be fired to the application.

The "connectionId" and "conferenceId" parameters are not both required; however, atleast one is.

- To delete a single connection, only specify "connectionId".
- To close an entire conference and delete all connections within it, only specify "conferenceId".
- To remove a connection from a conference without deleting it, specify both

"connectionId" and "conferenceId".

Requirements

Metreos Framework version 1.1

Metreos Media Server Provider version 1.1

Metreos.Providers.MediaServer.MuteConferenceConnection Action

Mute an individual connection in a conference.

Operation

Synchronous

Required Parameters

Parameter Name	Data Type	Description
conferenceId	System.Int32	Unique Media Server ID for a conference
connectionId	System.Int32	Unique Media Server ID for a connection

Optional Parameters

None

Result Data Fields

Parameter Name	Data Type	Description
resultCode	System.String	Result of a Media Server operation

Return Value

Returns "success" if the connection was muted; otherwise, "failure" is returned. If the media server operation fails the "resultCode" parameter will hold an integer value indicating the specific reason for the failure.

Remarks

The "MuteConferenceConnection" action allows applications to individually mute specific connections within a conference. Functionally, the media server sets the connection's mode to "receive only" allowing the connection to continue to receive audio, but audio received from that connection is ignored.

Requirements

Metreos Framework version 1.1

Metreos Media Server Provider version 1.1

Metreos.Providers.MediaServer.PlayAnnouncement Action

Play an audio file announcement to a specific media server connection. Optionally, the announcement may be played to all connections currently in a conference.

Operation

Asynchronous

Required Parameters

Parameter Name	Data Type	Description
userData	System.String	An opaque token used to correlate asynchronous actions with final responses in an application. Leave this set to the default 'none' unless you need to change it.

Optional Parameters

Parameter Name	Data Type	Description
audioFileAttribute	System.String	Attributes of the media file (i.e. format, encoding, bitrate) (deprecated)
audioFileBitrate	System.Int32	The bit rate of the media file
audioFileEncoding	System.String	The encoding of the media file ('ulaw' or 'alaw')
audioFileFormat	System.String	The format of the media file ('vox' or 'wav')
conferenceId	System.Int32	Unique Media Server ID for a conference
connectionId	System.Int32	Unique Media Server ID for a connection
filename	System.String	Name of the 1st file to play
filename2	System.String	Name of the 2nd file to play
filename3	System.String	Name of the 3rd file to play
state	System.String	User data which can be hair-pinned in a Media Server operation (deprecated)
termCondDigit	System.String	Digit to terminate the play announcement on.
termCondMaxDigits	System.Int32	Number of digits to receive before terminating the play announcement.
termCondMaxTime	System.Int32	Amount of time in milliseconds to wait before terminating the play announcement. Amount of silence in milliseconds to observe before terminating the play announcement.
termCondNonSilence	System.Int32	Amount of non-silence in milliseconds to observe before terminating the play announcement.
termCondSilence	System.Int32	Amount of silence in milliseconds to observe before terminating the command
terminationCondition	System.String	Condition under which the operation should complete (deprecated)

Result Data Fields

None.

Return Value

Returns "success" if the announcement has begun playing; otherwise, "failure" is returned. If the media server operation fails the "resultCode" parameter will hold an integer value indicating the specific reason for the failure.

Once the announcement starts playing the application will be notified that it is finished playing upon receipt of a "PlayAnnouncement_Complete" event.

Remarks

Announcements may be played to either a single connection or an entire conference on the media server. The media server will always stop playing an announcement once the end of the announcement file has been reached; however, the application may modify this behavior by adding additional termination conditions.

Termination conditions may be specified to stop the play announcement if a specific DTMF digit is received, if a certain amount of silence is observed, after a specific period of time, or if a specific number of DTMF digits are received. Some valid termination condition examples are:

- maxtime 3000
- digit #
- maxdigits 3
- silence 60000

NOTE: The "terminationCondition" parameter has been deprecated and is only present for backwards compatibility. Use the individual termination condition parameters such as "termCondDigit" for new application development.

The "connectionId" parameter is required only if "conferenceId" is not specified. If both parameters are specified then "connectionId" will take precedence.

The "filename" parameter refers to a file on the media server machine. The value of the parameter should be a relative path and filename to the announcement file to be played.

Playing audio files that are not recorded in VOX format requires the application to specify an "audioFileAttribute" specifying the file type of the announcement file. Currently, the only other supported file format is WAV. To play WAV files, specify "format wav" in the "audioFileAttribute" parameter. However, this is not required if the application is playing back an announcement that was previously recorded by the media server during a "RecordAudio" action. Note, WAV files should be recorded in PCM, 8kHz, 8bit Mono format.

NOTE: The "audioFileAttribute" parameter has been deprecated and is only present for backwards compatibility. Use the individual audio file attribute parameters such as "audioFileFormat" for new application development.

Requirements

Metreos Framework version 1.1

Metreos Media Server Provider version 1.1

Metreos.Providers.MediaServer.ReceiveDigits Action

Indicates to the media server that it should begin to monitor the media stream of a particular connection for DTMF digits.

Operation

Asynchronous

Required Parameters

Parameter Name	Data Type	Description
connectionId	System.Int32	Unique Media Server ID for a connection
terminationCondition	System.String	Condition under which the operation should complete
userData	System.String	An opaque token used to correlate asynchronous actions with final responses in an application. Use the default value of 'none' unless you need to change this.

Optional Parameters

Parameter Name	Data Type	Description
termCondDigit	System.String	A digit to observe that will terminate the command
termCondDigitList	System.Int32	A list of digits to observe that will terminate the command
termCondDigitPattern	System.Int32	A specific pattern of digits to observe that will terminate the command
termCondMaxDigits	System.Int32	Number of digits to receive before terminating the command
termCondMaxTime	System.Int32	Amount of time in milliseconds to wait before terminating the command
terminationCondition	System.String	Condition under which the operation should complete (deprecated)
state	System.String	Optional user state information to be returned when asynchronous command completes.

Result Data Fields

None.

Return Value

Returns "success" if the media server has begun to watch for DTMF digits; otherwise "failure" is returned.

When the media server stops watching for DTMF digits on the connection's media stream a "ReceiveDigits_Complete" event will be fired to the application. If an error occurs while processing the digits a "ReceiveDigits_Failed" event will be fired to the application.

Remarks

When receiving digits the media server continuously checks the digits received against the originally specified termination conditions. For example, termination conditions for the "ReceiveDigits" action could be to terminate on a specific number of digits received or when a specific digit (such as '#') is received.

Termination conditions may be specified to stop the receive digits if a specific DTMF digit is received, if a specific list of DTMF digits is received, after a specific period of time without receiving any DTMF digits, or if a specific number of DTMF digits are received.

NOTE: The "terminationCondition" parameter has been deprecated and is only present for backwards compatibility. Use the individual termination condition parameters such as "termCondDigit" for new application development.

The "ReceiveDigits" action will terminate when one of the specified termination conditions are observed. When this occurs, a "ReceiveDigits_Complete" event will fire to the application containing a parameter with the digits that were received.

Requirements

Metreos Framework version 1.1

Metreos Media Server Provider version 1.1

Metreos.Providers.MediaServer.RecordAudio Action

Records audio received on a specific media server connection. Optionally, record the mixed audio produced by a conference.

Operation

Asynchronous

Required Parameters

Parameter Name	Data Type	Description
userData	System.String	An opaque token used to correlate asynchronous actions with final responses in an application. Leave this defaulted to 'none' unless you need to change this.

Optional Parameters

Parameter Name	Data Type	Description
audioFileAttribute	System.String	Attributes of the media file (i.e. format, encoding, bitrate)
audioFileBitrate	System.Int32	The bit rate of the media file
audioFileEncoding	System.String	The encoding of the media file ('ulaw' or 'alaw')
audioFileFormat	System.String	The format of the media file ('vox' or 'wav')
commandTimeout	System.Int32	Max time to record (ms)
conferenceId	System.Int32	Unique Media Server ID for a conference
connectionId	System.Int32	Unique Media Server ID for a connection
expires	System.Int32	Amount of time in days to keep recorded file on server
filename	System.String	Name of the file to create containing the recorded audio
state	System.String	User data which can be hair-pinned in a Media Server operation (deprecated)
termCondDigit	System.String	A digit to observe that will terminate the command
termCondMaxTime	System.Int32	Amount of time in milliseconds to wait before terminating the command
termCondNonSilence	System.Int32	Amount of non-silence in milliseconds to observe before terminating the command
termCondSilence	System.Int32	Amount of silence in milliseconds to observe before terminating the command
terminationCondition	System.String	Condition under which the operation should complete

Result Data Fields

Parameter Name	Data Type	Description

connectionId	System.Int32	Id of the session

Return Value

Returns “success” if the media server has begun to record audio; otherwise “failure” is returned.

When the media server stops recording a “RecordAudio_Complete” event will be fired to the application. If an error occurred while recording, a “RecordAudio_Failed” event will be fired instead.

Remarks

Audio may be recorded from either a single connection or an entire conference on the media server. The media server will record audio until one of the specified termination conditions is satisfied.

Termination conditions may be specified to stop the record audio if a specific DTMF digit is received, if a certain amount of silence is observed, after a specific period of time, or if a certain amount of non-silence is observed.

NOTE: The ‘terminationCondition’ parameter has been deprecated and is only present for backwards compatibility. Use the individual termination condition parameters such as ‘termCondDigit’ for new application development.

The application may specify the file to be recorded to. Alternatively, if no “filename” parameter is specified then the media server will automatically generate a random file name. The file name that was recorded will be returned to the application when the final asynchronous event, either “RecordAudio_Complete” or “RecordAudio_Failed”, is fired to the application.

The application may also specify the format of the recording, encoding of the recorded file, and bit rate to record at. Furthermore, the “expires” parameter specifies the amount of time in days that the media server should retain the recorded file.

NOTE: The ‘audioFileAttribute’ parameter has been deprecated and is only present for backwards compatibility. Use the individual audio file attribute parameters such as ‘audioFileFormat’ for new application development.

The “connectionId” parameter is required only if “conferenceId” is not specified. If both parameters are specified then “connectionId” will take precedence.

Requirements

Metreos Framework version 1.1

Metreos Media Server Provider version 1.1

Metreos.Providers.MediaServer.SendDigits Action

Inserts digits into the processing stream of the media server.

Operation

Synchronous

Required Parameters

Parameter Name	Data Type	Description
connectionId	System.Int32	Unique Media Server ID for a connection
digits	System.String	A string of DTMF digits

Optional Parameters

None

Result Data Fields

None.

Return Value

Returns "success" if the digits were successfully inserted; otherwise, "failure" is returned.

Remarks

Applications sometimes receive digits from out-of-band sources (i.e., from the signaling path rather than media stream). When this happens it is necessary to insert those digits into the media stream to allow the media server to match those digits against existing termination conditions. "SendDigits" allows for this to happen, essentially enabling applications to proxy out-of-band DTMF digits back to the media server for processing.

Typically, this action would be used in response to the call control event "Metreos.CallControl.GotDigits" which fires when out-of-band digits are received. After receiving that event invoke the "SendDigits" action and set the "digits" parameter to the value of the parameter with the same name from the "Metreos.CallControl.GotDigits" event.

Requirements

Metreos Framework version 1.1

Metreos Media Server Provider version 1.1

Metreos.Providers.MediaServer.StopMediaOperation Action

Stop an executing, asynchronous media server operation on a specific connection to the media server.

Operation

Synchronous

Required Parameters

Parameter Name	Data Type	Description
connectionId	System.Int32	Unique Media Server ID for a connection

Optional Parameters

None

Result Data Fields

None.

Return Value

Returns "success" if operation was stopped; otherwise, "failure" is returned.

Remarks

If a media server connection is currently executing an asynchronous media server operation such as "PlayAnnouncement" the application may interrupt that operation by issuing the "StopMediaOperation". When interrupted a final asynchronous event will be fired to the application. The asynchronous event that is fired will contain a "reason" parameter whose value will be "userstop" to indicate that the command was terminated early at the request of the user.

Requirements

Metreos Framework version 1.1

Metreos Media Server Provider version 1.1

Metreos.Providers.MediaServer.UnMuteConferenceConnection Action

Un-mutes a previously muted connection in a conference.

Operation

Synchronous

Required Parameters

Parameter Name	Data Type	Description
conferenceId	System.Int32	Unique Media Server ID for a conference
connectionId	System.Int32	Unique Media Server ID for a connection

Optional Parameters

None

Result Data Fields

None.

Return Value

Returns “success” if the connection was un-muted; otherwise, “failure” is returned.

Remarks

The “UnMuteConferenceConnection” action allows applications to individually un-mute specific connections within a conference. Functionally, the media server sets the connection’s mode to “send and receive” allowing the connection to continue to receive audio and allowing audio received from that connection to be mixed into the conference.

Requirements

Metreos Framework version 1.1

Metreos Media Server Provider version 1.1

Asynchronous Callback Events

Metreos Framework Package Library

Metreos.Providers.MediaServer.PlayAnnouncement_Complete Event

Event fired indicating successful completion of a "PlayAnnouncement" action.

Type

Asynchronous Callback

Event Parameters

Parameter Name	Data Type	Description
resultCode	System.Int32	Result of a Media Server operation
state	System.String	User data which can be hair-pinned in a Media Server operation (deprecated)
terminationCondition	System.String	Condition under which the operation completed
userData	System.String	An opaque token used to correlate asynchronous actions with final responses in an application

Remarks

Upon receiving this event the "terminationCondition" parameter contains the reason that the media server stopped playing audio. The "terminationCondition" parameter can be one of the following:

- digit
- eod
- maxdigits
- maxtime
- silence
- userstop

Requirements

Metreos Framework version 1.1

Metreos Media Server Provider version 1.1

Metreos.Providers.MediaServer.PlayAnnouncement_Failed Event

Event fired indicating failure of a “PlayAnnouncement” action.

Type

Asynchronous Callback

Event Parameters

Parameter Name	Data Type	Description
resultCode	System.Int32	Result of a Media Server operation
state	System.String	User data which can be hair-pinned in a Media Server operation (deprecated)
userData	System.String	An opaque token used to correlate asynchronous actions with final responses in an application

Remarks

An error occurred which caused the “PlayAnnouncement” action to terminate prematurely without matching a specific termination condition.

Requirements

Metreos Framework version 1.1

Metreos Media Server Provider version 1.1

Metreos.Providers.MediaServer.ReceiveDigits_Complete Event

Event fired indicating successful completion of a "ReceiveDigits" action.

Type

Asynchronous Callback

Event Parameters

Parameter Name	Data Type	Description
state	System.String	User data which can be hair-pinned in a Media Server operation (deprecated)
digits	System.String	A string of DTMF digits
resultCode	System.Int32	Result of a Media Server operation
terminationCondition	System.String	Condition under which the operation completed
userData	System.String	An opaque token used to correlate asynchronous actions with final responses in an application

Remarks

Upon receiving this event the "digits" parameter will contain all digits received by the media server for the given connection ID. The "terminationCondition" parameter contains the reason that the media server stopped receiving digits. The "terminationCondition" parameter can be one of the following:

- digit
- digitlist
- digitpattern
- eod
- maxdigits
- maxtime
- userstop

Requirements

Metreos Framework version 1.1

Metreos Media Server Provider version 1.1

Metreos.Providers.MediaServer.ReceiveDigits_Failed Event

Event fired indicating failure of a “ReceiveDigits” action.

Type

Asynchronous Callback

Event Parameters

Parameter Name	Data Type	Description
state	System.String	User data which can be hair-pinned in a Media Server operation (deprecated)
resultCode	System.Int32	Result of a Media Server operation
userData	System.String	An opaque token used to correlate asynchronous actions with final responses in an application

Remarks

An error occurred which caused the “ReceiveDigits” action to terminate prematurely without matching a specific termination condition.

Requirements

Metreos Framework version 1.1

Metreos Media Server Provider version 1.1

Metreos.Providers.MediaServer.RecordAudio_Complete Event

Event fired indicating successful completion of a "RecordAudio" action.

Type

Asynchronous Callback

Event Parameters

Parameter Name	Data Type	Description
filename	System.String	Name of the file created containing the recorded audio
resultCode	System.Int32	Result of a Media Server operation
terminationCondition	System.String	Condition under which the operation completed
state	System.String	User data which can be hair-pinned in a Media Server operation (deprecated)
userData	System.String	An opaque token used to correlate asynchronous actions with final responses in an application

Remarks

Upon receiving this event the "filename" parameter will contain the name of the file that the audio was recorded to. The "terminationCondition" parameter contains the reason that the media server stopped recording audio. The "terminationCondition" parameter can be one of the following:

- digit
- eod
- maxtime
- nonsilence
- silence
- userstop

Requirements

Metreos Framework version 1.1

Metreos Media Server Provider version 1.1

Metreos.Providers.MediaServer.RecordAudio_Failed Event

Event fired indicating failure of a “RecordAudio” action.

Type

Asynchronous Callback

Event Parameters

Parameter Name	Data Type	Description
resultCode	System.Int32	Result of a Media Server operation
state	System.String	User data which can be hair-pinned in a Media Server operation (deprecated)
userData	System.String	An opaque token used to correlate asynchronous actions with final responses in an application

Remarks

An error occurred which caused the “RecordAudio” action to terminate prematurely without matching a specific termination condition.

Requirements

Metreos Framework version 1.1

Metreos Media Server Provider version 1.1

3. HTTP

The HTTP Provider supports the concept of sessions. A session is created the first time a script instance uses a Metreos.Providers.Http.SendResponse action. The HTTP Provider then will proxy any session-bound HTTP request to the script instance which created the session. A request to the HTTP Provider can signal that it is communicating to a session via one of two mechanisms:

- A header in the incoming request with a header named 'Metreos-SessionID', containing the Session GUID as the value.
- A cookie with name: 'metreosSessionId', and value of the Session GUID.

The client making these requests will be able to determine the Session GUID via one of two mechanisms:

- A header named 'Metreos-SessionID' containing the Session GUID as a value is present in all 200 OK responses from the Application Server, which an intelligent client can then save and use for subsequent requests.
- For support of most browser-type clients, a Set-Cookie header will be sent in a response to the client in a properly formed Metreos.Providers.Http.SendResponse. The Cookie will have the name 'metreosSessionId', and the value will be the Session GUID.

The Session GUID is the same value as the routing guid of the script instance.

The application developer should supply a Set-Cookie header in a SendResponse action when dealing with browsers or Cisco IP Phones, so that it will pass the cookie in all subsequent requests to gain access to any non-triggering Metreos.Provider.Http.GotRequest event handlers contained in this script.

Sessions fire an event after a configurable amount of time after initialization to the script instance. This event, SessionExpired, can then be handled however the application developer sees fit.

Actions

Metreos Framework Package Library

Metreos.Providers.Http.SendResponse Action

Send an HTTP response to a request that was previously received.

Operation

Synchronous

Required Parameters

Parameter Name	Data Type	Description
remoteHost	System.String	Address of host who sent the original request
responseCode	System.Int32	Numeric code of HTTP response

Optional Parameters

Parameter Name	Data Type	Description
body	System.String	Body of the response
responsePhrase	System.String	Description of response
Content-Type	System.String	Any number of headers in the HTTP response

Result Data Fields

None

Return Value

Returns “success” if the response was sent; otherwise, “failure” is returned.

Remarks

When sending responses to HTTP requests the application must specify, at a minimum, “responseCode” and “remoteHost”. The “remoteHost” parameter is used by the HTTP Provider to correlate this response with the originating HTTP request. The application should set the value of the “remoteHost” parameter to the value of the parameter of the same name received in the “GotRequest” event.

Optionally, an application may specify any other headers as parameters in the “SendResponse” action. For example, if the application wanted to specify the content type of the response body, a parameter named “content-type” should be inserted into the action with the appropriate value.

Requirements

Metreos Framework version 1.1

Metreos HTTP Provider version 1.1

Events

Metreos Framework Package Library

Metreos.Providers.Http.GotRequest Event

Event fired when an HTTP request is received by the application server.

Type

Hybrid

Event Parameters

Parameter Name	Data Type	Description
host	System.String	Host portion of the request URI (may include port info)
hostname	System.String	Host portion of the request URI (no port info)
method	System.String	HTTP method (POST, GET, etc)
port	System.Int32	Port portion of the request URI
query	System.String	Query string portion of the request URI
remoteHost	System.String	The IP address and port of the remote client
remoteIpAddress	System.String	The IP address of the remote client
url	System.String	Path portion of the request URI
[headers]	System.String	Any number of headers in the HTTP request

Remarks

This event will fire to the application server when either a GET or POST request arrives at the application server. All HTTP headers will be maintained and can be access from the event parameters by using the header name as the parameter name.

Requirements

Metreos Framework version 1.1

Metreos HTTP Provider version 1.1

Metreos.Providers.Http.SessionExpired Event

Event fired when an HTTP request is received by the application server.

Type

Non-triggering

Event Parameters

Remarks

The HTTP Provider regularly checks all outstanding sessions to determine if any have expired. If a session is found to have expired, then this event is sent to the script instance that was started when the session was created.

Requirements

Metreos Framework version 1.1

Metreos HTTP Provider version 1.1

Types

Metreos Framework Package Library

Metreos.Types.Http.QueryParamCollection Type

Parses the query parameter section of an HTTP URI, populating a collection with key-value pairs. The collection can then be used to determine which query parameters came in with an HTTP request.

Serializable

Yes

Input Types

Type Name	Description
string	The section of the URI containing the query parameters, exclusive or inclusive of the beginning '?'.

Accessible Public Methods

Method Name	Description
string this[string name]	Returns the value of the named query parameter. An empty string is a valid value for a query parameter. If a query parameter was not defined in the supplied query parameter URI fragment to parse, then it will return as null.
string this[int index]	Returns the value of the named query parameter. An empty string is a valid value for a query parameter. If a query parameter was not defined in the supplied query parameter URI fragment to parse, then it will return as null.
string GetNameAt(int index)	Returns the name of the query parameter at the given index.
void Clear()	Clears the collection.

Accessible Public Properties

Property Name	Description
int Count	Returns the number of query parameters in the collection.

Remarks

The most commonly intended use of this type is to initialize it with the 'query' event parameter which comes in with a Metreos.Providers.Http.GotRequest.

Requirements

Metreos Framework version 1.1

4. APPLICATION CONTROL

Actions

Metreos Framework Package Library

Metreos.ApplicationControl.SetSessionData Action

Sets the value of a session data variable.

Operation

Synchronous

Required Parameters

None.

Optional Parameters

Parameter Name	Data Type	Description
[parameters]	System.Object	Parameters that will be stored in the session data hashtable.

Result Data Fields

None.

Return Value

"success" if the modification was successful, otherwise "failure"

Remarks

Any number of SessionData entries can be specified. The name of a defined parameter will be used as a key into the "sessionData.customData" hashtable, and the value specified for the defined parameter will be used as the value for the above-mentioned key. For example, if a parameter *'a'* is created, and the literal string *'true'* assigned to it, '*true'* will be stored in the sessionData.customData hashtable with *'a'* as the key, and *'true´* as the value. In order to access session data in a custom code action, '*SessionData sessionData'* needs to be specified as a method parameter. Since the value is stored as an object inside the hashtable, you may have to cast any retrieved elements to the proper type before using them.

Requirements

Metreos Framework version 1.1

Metreos.ApplicationControl.EnableScript Action

Enables a slave script for execution.

Operation

Synchronous

Required Parameters

Parameter Name	Data Type	Description
ScriptName	System.String	Name of the script that you wish to enable.

Optional Parameters

Parameter Name	Data Type	Description
[parameters]	System.Object	Triggering parameters for the script that you wish to enable.

Result Data Fields

None.

Return Value

"success" if the operation was successful, otherwise "failure"

Remarks

This action allows you to enable a slave script from a running application. You need to specify the name of the script that you wish to start, as well as all the required triggering parameters for that script. The names of the parameters you specify in this action need to match the names of the triggering parameters of the target script.

Requirements

Metreos Framework version 1.1

Metreos.ApplicationControl.CallFunction Action

Calls a function inside the currently executing application.

Operation

Synchronous

Required Parameters

Parameter Name	Data Type	Description
functionName	System.String	The name of the function to invoke

Optional Parameters

Parameter Name	Data Type	Description
[parameters]	System.String	Any number of parameters that can be passed in to the invoked function.

Result Data Fields

Parameter Name	Data Type	Description
[resultDataFields]	System.String	Any number of result data fields passed back from the exited invoked function

Return Value

The return value of this action is specified by the function that is being called.

Remarks

Well designed applications often have repeated logic that can be refactored into a single function. This action allows applications to define stand-alone, non-event handling functions and to invoke those functions from their application logic. However, this action may be used to call event-handling functions as well as long as the "CallFunction" action contains all required parameters expected by the event handler.

Requirements

Metreos Framework version 1.1

Metreos.ApplicationControl.EndScript Action

Exit and cleanup the currently executing application instance.

Operation

Synchronous

Required Parameters

None

Optional Parameters

None

Result Data Fields

None

Return Value

None.

Remarks

Application instances will continue to run until an "EndScript" action is processed. This action will inform the application server that the currently executing application instance is ready to be shut down and cleaned up. After processing an "EndScript" action no further events will be received by the application instance and no further actions may be invoked by the application instance.

Requirements

Metreos Framework version 1.1

Metreos.ApplicationControl.EndFunction Action

Exits the currently executing function.

Operation

Synchronous

Required Parameters

None

Optional Parameters

Parameter Name	Data Type	Description
ReturnValue	System.String	The value used as the branching condition of the CallFunction which invoked the current function
[resultDataFields]	System.String	Any number of result data fields returned to the CallFunction which invoked the function

Result Data Fields

None

Return Value

None.

Remarks

Functions may exit at any time. If a function for an event handler exits, then the application instance will continue to run waiting for further events. When a function exits after being called by another function, execution of the application returns to the calling function.

Functions may return any number of result data fields to the calling function by adding parameters to the "EndFunction" action. Any parameters present in the action will be passed to the calling function for processing.

Return values may be specified in the "ReturnValue" parameter. The return value allows the calling function to branch on the success or failure of the function that was called.

Requirements

Metreos Framework version 1.1

Metreos.ApplicationControl.Forward Action

Forwards all subsequent events for a particular application script type to another application script type.

Operation

Synchronous

Required Parameters

Parameter Name	Data Type	Description
ToGuid	System.String	The routing GUID of the running application in which to send the event

Optional Parameters

None

Result Data Fields

None

Return Value

None.

Remarks

This action instructs the application server to forward all future events and responses to another application. It requires a "toGuid" parameter which holds the routing GUID of the target application. In order to get that information, the target application would have to make it available via the registry, a database, or some other thread-safe mechanism. Once this action is executed, no further actions are taken and the application instance is reset.

Requirements

Metreos Framework version 1.1

Metreos.ApplicationControl.SendEvent Action

Fires an event to another application.

Operation

Synchronous

Required Parameters

Parameter Name	Data Type	Description
EventName	System.String	The name for the event

Optional Parameters

Parameter Name	Data Type	Description
ToGuid	System.String	The routing GUID of the running application in which to send the event
[parameters]	System.String	Any number of parameters to pass into the event

Result Data Fields

None

Return Value

Returns “success” if the event was fired; otherwise, “failure” is returned.

Remarks

This action enables application scripts to communicate with other application scripts installed on the application server. The “eventName” parameter specifies the name of the event that will be fired. The “toGuid” parameter specifies the routing GUID of a running application script. If the “toGuid” parameter is not specified, the event will be handled like any other triggering event.

Applications may specify any number of parameters in the “SendEvent” action. These parameters will be preserved and passed to the application instance receiving the event.

Requirements

Metreos Framework version 1.1

Metreos.Native.Log.Write Action

Outputs a message to the application's log.

Operation

Synchronous

Required Parameters

Parameter Name	Data Type	Description
LogLevel	System.String	The trace level of the log message
Message	System.String	The log message

Optional Parameters

Parameter Name	Data Type	Description
ApplicationName	System.String	The name to report as the source of the message

Result Data Fields

None

Return Value

Returns "success" if the log message was output; otherwise, "failure" is returned.

Remarks

Applications may use this action to write informative log messages to the application log for debugging and other purposes. Log messages output by applications will be processed by the application server's log subsystem and potentially output to multiple locations (console, file, Windows event log, etc).

The "logLevel" parameter specifies the priority of the log message. It may be either "Verbose", "Info", "Warning", or "Error". This parameter is used by the application server to effectively filter the log messages for various output devices.

Requirements

Metreos Framework version 1.1

5. CONDITIONALS

This set of actions allows for testing of and branching on user-defined conditions. The general procedure for using these actions is to drag one onto the canvas, specify the Boolean condition in one of the value fields, and then specify connections to other nodes using branch conditions specific to each action. The “success” and “failure” branch conditions do not apply to conditional actions.

Actions

Metreos Framework Package Library

Metreos.Native.Conditional.If Action

Returns “true” if the specified condition is true, returns “false” otherwise.

Operation

Synchronous

Required Parameters

Parameter Name	Data Type	Description
Value1	System.Boolean	The value on which to perform the check.

Optional Parameters

None

Result Data Fields

None

Return Value

Returns “true” or “false”, depending on what Value1 evaluated to.

Remarks

This action takes a statement that evaluates to either *true* or *false*. If the evaluated statement is *true*, the action returns “true” as its return value. If the evaluated statement is false, the action returns “false” In order to branch on this event, the branch conditions must be set to one of the following: true, false, or default.

Requirements

Metreos Framework version 1.1

Metreos.Native.Conditional.Switch Action

Allows branching on a user defined object.

Operation

Synchronous

Required Parameters

Parameter Name	Data Type	Description
SwitchOn	System.Object	The value on which to perform the switch.

Optional Parameters

None

Result Data Fields

None

Return Value

Returns SwitchOn.ToString(), or “null” if SwitchOn is null.

Remarks

This action simply returns the value of the method ToString() on whatever object was passed as the SwitchOn argument. This allows the application developer to specify custom branch conditions. For example, if one passes a string variable into this action, one can specify branching conditions such as “bob” or “tom”. These branches will be taken if they match the value of SwitchOn.ToString(). Usage of a “default” branch is recommended in case none of the user-specified branches succeed.

Requirements

Metreos Framework version 1.1

Metreos.Native.Conditional.Compare Action

Compares two values to each other.

Operation

Synchronous

Required Parameters

Parameter Name	Data Type	Description
Value1	System.Object	The first value.
Value2	System.Object	The second value.

Optional Parameters

None

Result Data Fields

None

Return Value

Returns “equal” if the two values are equal to each other, and it returns “unequal” if they are not.

Remarks

None.

Requirements

Metreos Framework version 1.1

6. DATABASE

The database actions allow the application to interface with an external database server such as Oracle or MySQL.

Actions

Metreos Framework Package Library

Metreos.Native.Database.OpenDatabase Action

Compares two values to each other.

Operation

Synchronous

Required Parameters

Parameter Name	Data Type	Description
DSN	System.String	The DSN string for the desired database.
Name	System.String	Name of the connection.

Optional Parameters

Parameter Name	Data Type	Description
Type	System.String	The type of the database connection.

Result Data Fields

None

Return Value

Returns “success” if the database connection was successful, returns “failure” if it was not.

Remarks

The *DSN* string specifies the information required by the action in order to find the database and connect to it. *Name* assigns a name to the connection, and can later be used by other database actions to refer to this specific connection. Valid values for Type are oracle, sqlserver, or odbc.

Requirements

Metreos Framework version 1.1

Metreos.Native.Database.ExecuteQuery Action

Compares two values to each other.

Operation

Synchronous

Required Parameters

Parameter Name	Data Type	Description
Query	System.String	The query you want to execute on the database server.
Name	System.String	Name of the connection on which to perform the query.

Optional Parameters

None

Result Data Fields

Parameter Name	Data Type	Description
ResultSet	System.Data. DataTable	The result of the query.

Return Value

Returns “success” if the query executed successfully, returns “failure” if it did not.

Remarks

Name specifies the name of a connection that was earlier created using the *OpenDatabase* action. *Query* is a SQL statement that you want executed on the database. The values returned from the query may be placed into a variable of type System.Data.DataTable.

Requirements

Metreos Framework version 1.1

Metreos.Native.Database.ExecuteCommand Action

Compares two values to each other.

Operation

Synchronous

Required Parameters

Parameter Name	Data Type	Description
Command	System.String	The command you want to execute on the database server.
Name	System.String	Name of the connection on which to perform the query.

Optional Parameters

None

Result Data Fields

Parameter Name	Data Type	Description
RowsAffected	System.Int32	The number of rows affected by the command.

Return Value

Returns “success” if the command executed successfully, returns “failure” if it did not.

Remarks

Name specifies the name of a connection that was earlier created using the *OpenDatabase* action. *Command* is a SQL statement that you want executed on the database.

Requirements

Metreos Framework version 1.1

7. CISCO IP PHONE

This set of native actions provides utilities for the creation of complex displays for Cisco IP phones. The general procedure for creating a display is to declare a variable of the appropriate type and then execute the corresponding "Create..." action to fill in the basic details. Next, one or more "Add..." actions can be executed to add various components to the display such as soft keys and icons.

Once the variable has been initialized and populated with all desired components, it may be sent in the body of a 200 OK response to a request from the phone, accompanied with a ‘text/xml’ Content-Type. The one exception to this rule is an Execute object which is created in the same manner, but is transported in the body of an HTTP "PUT" request by way of the “SendExecute” command.

Actions

Metreos Framework Package Library

Metreos.Native.CiscoIpPhone.AddDirectoryEntry Action

Add an entry to a Cisco IP Phone directory.

Operation

Synchronous

Required Parameters

None

Optional Parameters

Parameter Name	Data Type	Description
Name	System.String	Name of the directory entry
Telephone	System.String	Telephone number of the directory entry

Result Data Fields

Parameter Name	Data Type	Description
ReturnValue	System.String	Conforms to the XML representing a directory entry in a Metreos.Types.CiscoIpPhone.Directory type

Return Value

Returns a properly formatted Cisco IP Phone directory entry. The "returnValue" parameter should be assigned to the "Metreos.Types.CiscoIpPhone.Directory" variable that the application wishes to add an entry to.

Remarks

When adding directory entries to a Cisco IP Phone directory the application should define a variable of type "Metreos.Types.CiscoIpPhone.Directory". Then the application should assign the "returnValue" parameter of the "AddDirectoryEntry" action to the variable that is holding the Cisco IP Phone directory. Doing this allows the directory object to insert the entry created by the "AddDirectoryEntry" action.

Requirements

Metreos Framework version 1.1

Metreos HTTP Provider version 1.1

Metreos.Native.CiscoIpPhone.AddIconItem Action

Adds an icon item to an icon menu.

Operation

Synchronous

Required Parameters

None

Optional Parameters

Parameter Name	Data Type	Description
Data	System.String	Byte array of the data comprising the icon
Depth	System.UInt16	The number of bits used to determine the colors available for the icon
Height	System.UInt16	The height of the icon (pixels)
Index	System.UInt16	The index of the icon. This value is referenced by the AddMenuItem action
Width	System.UInt16	The width of the icon (pixels)

Result Data Fields

Parameter Name	Data Type	Description
ResultData	CiscoIpPhone. CiscoIPPhoneInput ItemType	Conforms to the XML representing an icon item in a Metreos.Types.CiscoIpPhone.IconMenu type

Return Value

Returns a properly formatted Cisco IP Phone icon item entry.

Remarks

Icons may be added to Cisco IP Phone icon menus. The icon only needs to be added once and is identified by its "Index" parameter. When referencing this icon later when adding menu items, refer to the value of the "Index" parameter.

The "Data" parameter of the "AddIconItem" action must be a properly formatted CIP image. Refer to the "Cisco IP Phone SDK" for more information on the CIP format.

When building a Cisco IP Phone icon menu, the application should define a variable of type "Metreos.Types.CiscoIpPhone.IconMenu". Then the application should assign the "ResultData" parameter of the "AddIconItem" action to the variable that is holding the Cisco IP Phone icon menu. Doing this allows the icon menu object to insert the entry created by the "AddIconItem" action.

Requirements

Metreos Framework version 1.1

Metreos.Native.CiscoIpPhone.AddInputItem Action

Adds an input field to an input object.

Operation

Synchronous

Required Parameters

None

Optional Parameters

Parameter Name	Data Type	Description
DefaultValue	System.String	The value to initially display for the input item
DisplayName	System.String	The name of the input
InputFlags	System.String	Indicates the behavior of the input item
QueryStringParam	System.String	The name of the query parameter that will be sent in the HTTP GET executed by a Cisco IP Phone, if the input item is selected

Result Data Fields

Parameter Name	Data Type	Description
ResultData	Metreos.Types.CiscoIpPhone.CiscoIPPhoneInputItemType	Conforms to the XML representing an input item in a Metreos.Types.CiscoIpPhone.Input type

Return Value

Returns a properly formatted Cisco IP Phone input item entry. The "ResultData" parameter should be assigned to the "Metreos.Types.CiscoIpPhone.Input" variable that the application wishes to add the input item to.

Remarks

The "InputFlags" parameter specifies the type of input field to generate. Valid input flags are:

- A – Plain ASCII text. Upper and lower case text consisting of letters, numbers and special characters.
- T – Telephony number. Only numbers, '#' and '*'.
- N – Numeric. Only numbers.
- E – Equation. Numbers and the special math symbols.
- U – Uppercase. Only uppercase letters.
- L – Lowercase. Only lowercase letters.
- P – Password. Individual characters display as they are entered and then change to an asterisk to mask the entered value.

When building a Cisco IP Phone input object, the application should define a

variable of type “Metreos.Types.CiscoIpPhone.Input”. Then the application should assign the “ResultData” parameter of the “AddInputItem” action to the variable that is holding the Cisco IP Phone input object. Doing this allows the input object to insert the entry created by the “AddInputItem” action.

Requirements

Metreos Framework version 1.1

Metreos HTTP Provider version 1.1

Metreos.Native.CiscoIpPhone.AddMenuItem Action

Adds a menu item to a menu object.

Operation

Synchronous

Required Parameters

None

Optional Parameters

Parameter Name	Data Type	Description
IconIndex	System.UInt16	If this action is appending to a Metreos.Types.CiscoIpPhone.IconMenu, then this value indicates which index to reference in displaying an icon
Name	System.String	Name of the menu item
TouchArea-X1	System.Int16	Represents the X-coordinate of the upper-left corner of a bounded area that will cause this menu item to be selected when touched (pixels) *touch screen enabled phones only*
TouchArea-X2	System.Int16	Represents the X-coordinate of the lower-right corner of a bounded area that will cause this menu item to be selected when touched (pixels) *touch screen enabled phones only*
TouchArea-Y1	System.Int16	Represents the Y-coordinate of the upper-left corner of a bounded area that will cause this menu item to be selected when touched (pixels) *touch screen enabled phones only*
TouchArea-Y2	System.Int16	Represents the Y-coordinate of the lower-right corner of a bounded area that will cause this menu item to be selected when touched (pixels) *touch screen enabled phones only*
URL	System.String	The URL to request if this particular menu item is selected

Result Data Fields

Parameter Name	Data Type	Description
ResultData	Metreos.Types.CiscoIpPhone. CiscoIPPhoneMenuItem Type	Conforms to the XML representing a menu item in a Metreos.Types.CiscoIpPhone.Menu, Metreos.Types.CiscoIpPhone.IconMenu, Metreos.Types.CiscoIpPhone.GraphicMenu, or a Metreos.Types.CiscoIpPhone.GraphicFileMenu type

Return Value

Returns a properly formatted Cisco IP Phone menu item entry. The "ResultData" parameter should be assigned to any menu type variable (listed above) that the

application wishes to add the input item to.

Remarks

The parameters of this action indicate display name, icon to use (if appropriate), link address, and placement of the menu item on the phone display. For touch screen enabled phones, a touch area can be designated for the item as well.

Requirements

Metreos Framework version 1.1

Metreos HTTP Provider version 1.1

Metreos.Native.CiscoIpPhone.AddSoftKeyItem Action

Adds a soft key item to almost any Cisco IP phone format type.

Operation

Synchronous

Required Parameters

None

Optional Parameters

Parameter Name	Data Type	Description
Name	System.String	Name of the soft key item
Position	System.UInt16	The position of the soft key item
URL	System.String	The URL to request if this particular soft key item is selected

Result Data Fields

Parameter Name	Data Type	Description
ResultData	Metreos.Types.CiscoIpPhone.CiscoIPPhoneSoftKeyType	Conforms to the XML representing a soft key item in a Metreos.Types.CiscoIpPhone.Text, Metreos.Types.CiscoIpPhone.Directory, Metreos.Types.CiscoIpPhone.Menu, Metreos.Types.CiscoIpPhone.Input, Metreos.Types.CiscoIpPhone.IconMenu, Metreos.Types.CiscoIpPhone.Image, Metreos.Types.CiscoIpPhone.ImageFile, Metreos.Types.CiscoIpPhone.GraphicMenu, or a Metreos.Types.CiscoIpPhone.GraphicFileMenu type

Return Value

Returns a properly formatted Cisco IP Phone soft key. The “ResultData” parameter should be assigned to almost any variable type (listed above) that the application wishes to add the soft key to.

Remarks

This action adds a soft key to any screen on the Cisco IP phone. Soft keys appear at the bottom of the screen and correspond to physical keys located immediately underneath the display. The “URL” parameter indicates where to phone should look for the screen to display when the defined soft key is pressed.

Requirements

Metreos Framework version 1.1

Metreos HTTP Provider version 1.1

Metreos.Native.CiscoIpPhone.CreateDirectory Action

Creates a directory display

Operation

Synchronous

Required Parameters

None

Optional Parameters

Parameter Name	Data Type	Description
Prompt	System.String	Prompt to display on the Cisco IP phone
Title	System.String	Title to display on the Cisco IP phone

Result Data Fields

Parameter Name	Data Type	Description
ResultData	Metreos.Types.CiscoIpPhone.CiscoIPPhoneDirectoryType	Represents a minimal CiscoIpPhone.Directory message. This value can then be aggregated by a AddDirectoryEntry and a AddSoftKeyItem

Return Value

Returns a properly formatted Cisco IP Phone directory display. The "ResultData" parameter should be assigned to a variable of type "Metreos.Types.CiscoIpPhone.DirectoryType".

Remarks

This action creates a directory object and fills in only the basic properties. Individual directory items can be added by calling "AddDirectoryEntry" and specifying the resultant directory object of this operation as the target of the "AddDirectoryEntry" action.

Requirements

Metreos Framework version 1.1

Metreos HTTP Provider version 1.1

Metreos.Native.CiscoIpPhone.CreateExecute Action

Creates an "execute" command which can be pushed to the phone.

Operation

Synchronous

Required Parameters

None

Optional Parameters

Parameter Name	Data Type	Description
URL1	System.String	Any valid URI
URL2	System.String	Any valid URI
URL3	System.String	Any valid URI
Priority1	System.UInt16	A valid priority
Priority2	System.UInt16	A valid priority
Priority3	System.UInt16	A valid priority

Result Data Fields

Parameter Name	Data Type	Description
ResultData	Metreos.Types.CiscoIpPhone.CiscoIPPhoneExecuteType	Represents a CiscoIpPhone.Execute message. This value cannot be further aggregated.

Return Value

Returns a properly formatted Cisco IP Phone execute command. The "ResultData" parameter should be assigned to a variable of type "Metreos.Types.CiscoIpPhone.Execute".

Remarks

This action creates a command which can be pushed to the phone by way of the "SendExecute" action. See the "Cisco IP Phone SDK" for more details on the valid URI schemes and priority codes supported by the Cisco IP Phones.

Requirements

Metreos Framework version 1.1

Metreos HTTP Provider version 1.1

Metreos.Native.CiscoIpPhone.CreateInput Action

Creates an input display.

Operation

Synchronous

Required Parameters

None

Optional Parameters

Parameter Name	Data Type	Description
Prompt	System.String	Prompt to display on the Cisco IP phone
Title	System.String	Title to display on the Cisco IP phone
URL	System.String	The base URL used when an input item is selected

Result Data Fields

Parameter Name	Data Type	Description
ResultData	Metreos.Types.CiscoIpPhone.CiscoIPPhoneInputType	Represents a minimal CiscoIpPhone.Input message. This value can then be aggregated by a AddInputItem and a AddSoftKeyItem

Return Value

Returns a properly formatted Cisco IP Phone input display. The "ResultData" parameter should be assigned to a variable of type Metreos.Types.CiscoIpPhone.Input.

Remarks

This action creates an input object and fills in only the basic properties. Individual input items can be added by calling "AddInputItem" and specifying the resultant input object of this operation as the target of the "AddInputItem" action.

Requirements

Metreos Framework version 1.1

Metreos HTTP Provider version 1.1

Metreos.Native.CiscoIpPhone.CreateIconMenu Action

Creates an icon menu display

Operation

Synchronous

Required Parameters

None

Optional Parameters

Parameter Name	Data Type	Description
Prompt	System.String	Prompt to display on the Cisco IP phone
Title	System.String	Title to display on the Cisco IP phone
URL	System.String	URL that the icon menu points to

Result Data Fields

Parameter Name	Data Type	Description
ResultData	Metreos.Types.CiscoIpPhone.CiscoIPPhoneInputType	Represents a minimal CiscoIpPhone.IconMenu message. This value can then be aggregated by a AddMenuItem, AddIconItem, and a AddSoftKeyItem

Return Value

Returns a properly formatted Cisco IP Phone icon menu display. The "ResultData" parameter should be assigned to a variable of type Metreos.Types.CiscoIpPhone.IconMenu.

Remarks

This action creates an icon menu object and fills in only the basic properties. Individual input items can be added by calling "AddMenuItem" or "AddIconItem" and specifying the resultant icon menu object of this operation as the target of the other actions.

Requirements

Metreos Framework version 1.1

Metreos HTTP Provider version 1.1

Metreos.Native.CiscoIpPhone.CreateImage Action

Creates an image display

Operation

Synchronous

Required Parameters

None

Optional Parameters

Parameter Name	Data Type	Description
Data	System.String	Byte array of the data comprising the image
Depth	System.UInt16	The number of bits used to determine the colors available for the image
Height	System.UInt16	The height of the image (pixels)
LocationX	System.Int16	The X-coordinate of the upper-left corner of the image to display (pixels)
LocationY	System.Int16	The Y-coordinate of the upper-left corner of the image to display (pixels)
Prompt	System.String	Prompt to display on the Cisco IP phone
Title	System.String	Title to display on the Cisco IP phone
Width	System.UInt16	The width of the image (pixels)

Result Data Fields

Parameter Name	Data Type	Description
ResultData	Metreos.Types.CiscoIpPhone.CiscoIPPhoneImageType	Represents a minimal CiscoIpPhone.Image message. This value can then be aggregated by a AddSoftKeyItem

Return Value

Returns a properly formatted Cisco IP Phone image display. The "ResultData" parameter should be assigned to a variable of type Metreos.Types.CiscoIpPhone.Image.

Remarks

This action creates a properly formatted image object to be displayed on the Cisco IP Phone.

The "Data" parameter of the "CreateImage" action must be a properly formatted CIP image. Refer to the "Cisco IP Phone SDK" for more information.

Requirements

Metreos Framework version 1.1

Metreos.Native.CiscoIpPhone.CreateImageFile Action

Creates a file image display (models 7970 or greater only)

Operation

Synchronous

Required Parameters

None

Optional Parameters

Parameter Name	Data Type	Description
LocationX	System.Int16	The X-coordinate of the upper-left corner of the image to display (pixels)
LocationY	System.Int16	The Y-coordinate of the upper-left corner of the image to display (pixels)
Prompt	System.String	Prompt to display on the Cisco IP phone
Title	System.String	Title to display on the Cisco IP phone
URL	System.String	URL of the PNG image to download to the phone

Result Data Fields

Parameter Name	Data Type	Description
ResultData	Metreos.Types. CiscoIpPhone. CiscoIPPhoneImageFileType	Represents a minimal CiscoIpPhone.ImageFile message. This value can then be aggregated by a AddSoftKeyItem

Return Value

Returns a properly formatted Cisco IP Phone file image display. The "ResultData" parameter should be assigned to a variable of type "Metreos.Types.CiscoIpPhone.ImageFile".

Remarks

This action creates a properly formatted "ImageFile" object to be displayed on a Cisco IP Phone.

This action creates a file image object. The "URL" parameter must point to a PNG image within the bounds of what is displayable on the phone.

Requirements

Metreos Framework version 1.1

Metreos HTTP Provider version 1.1

Metreos.Native.CiscoIpPhone.CreateGraphicMenu Action

Creates a graphic menu display

Operation

Synchronous

Required Parameters

None

Optional Parameters

Parameter Name	Data Type	Description
Data	System.String	Byte array of the data comprising the image
Depth	System.UInt16	The number of bits used to determine the colors available for the image
Height	System.UInt16	The height of the image (pixels)
LocationX	System.Int16	The X-coordinate of the upper-left corner of the image to display (pixels)
LocationY	System.Int16	The Y-coordinate of the upper-left corner of the image to display (pixels)
Prompt	System.String	Prompt to display on the Cisco IP phone
Title	System.String	Title to display on the Cisco IP phone
Width	System.UInt16	The width of the image (pixels)

Result Data Fields

Parameter Name	Data Type	Description
ResultData	Metreos.Types.CiscoIpPhone.CiscoIPPhoneGraphicMenuType	Represents a minimal Cisco IP Phone GraphicMenu message. This value can then be aggregated by a AddMenuItem and a AddSoftKeyItem

Return Value

Returns a properly formatted Cisco IP Phone graphic menu display. The "ResultData" parameter should be assigned to a variable of type "Metreos.Native.CiscoIpPhone.GraphicMenu".

Remarks

This action creates the graphic menu object and fills in only the basic properties (including image data). Individual menu items can be added by calling AddMenuItem and specifying the resultant menu object of this operation as the target of the AddMenuItem action.

The "Data" parameter of the "CreateGraphicMenu" action must be a properly formatted CIP image. Refer to the "Cisco IP Phone SDK" for more information on

the CIP format.

Requirements

Metreos Framework version 1.1

Metreos HTTP Provider version 1.1

Metreos.Native.CiscoIpPhone.CreateGraphicFileMenu Action

Creates a graphic file menu display (models 7970 or greater only)

Operation

Synchronous

Required Parameters

None

Optional Parameters

Parameter Name	Date Type	Description
LocationX	System.Int16	The X-coordinate of the upper-left corner of the image to display (pixels)
LocationY	System.Int16	The Y-coordinate of the upper-left corner of the image to display (pixels)
Prompt	System.String	Prompt to display on the Cisco IP phone
Title	System.String	Title to display on the Cisco IP phone
URL	System.String	URL of the PNG image to download to the phone

Result Data Fields

Parameter Name	Date Type	Description
ResultData	Metreos.Types.CiscoIpPhone. CiscoIPPhoneGraphicFileMenuType	Represents a minimal Cisco IP Phone graphic file menu message. This value can then be aggregated by a AddMenuItem and a AddSoftKeyItem

Return Value

Returns a properly formatted Cisco IP Phone graphic file menu display. The "ResultData" parameter should be assigned to a variable of type "Metreos.Types.CiscoIpPhone.GraphicFileMenu".

Remarks

This action creates a graphic menu object and fills in only the basic properties (including image data). Individual menu items can be added by calling "AddMenuItem" and specifying the resultant menu object of this operation as the target of the "AddMenuItem" action.

The "URL" parameter must point to a properly formatted PNG image.

Requirements

Metreos Framework version 1.1

Metreos HTTP Provider version 1.0

Metreos Framework Package Library

Metreos.Native.CiscoIpPhone.CreateMenu Action

Creates a menu display.

Operation

Synchronous

Required Parameters

None

Optional Parameters

Parameter Name	Data Type	Description
Prompt	System.String	Prompt to display on the Cisco IP phone
Title	System.String	Title to display on the Cisco IP phone

Result Data Fields

Parameter Name	Data Type	Description
ResultData	Metreos.Types.CiscoIpPhone.CiscoIPPhoneMenuType	Represents a minimal Cisco IP Phone menu message. This value can then be aggregated by a AddMenuItem and a AddSoftKeyItem

Return Value

Returns a properly formatted Cisco IP Phone menu display. The "ResultData" parameter should be assigned to a variable of type Metreos.Types.CiscoIpPhone.Menu.

Remarks

This action creates a menu object and fills in only the basic properties. Individual menu items can be added by calling "AddMenuItem" and specifying the resultant menu object of this operation as the target of the "AddMenuItem" action.

Requirements

Metreos Framework version 1.1

Metreos HTTP Provider version 1.1

Metreos.Native.CiscoIpPhone.CreateText Action

Creates a simple text display.

Operation

Synchronous

Required Parameters

None

Optional Parameters

Parameter Name	Data Type	Description
Prompt	System.String	Prompt to display on the Cisco IP phone
Text	System.String	The text to display on the Cisco IP phone
Title	System.String	Title to display on the Cisco IP phone

Result Data Fields

Parameter Name	Data Type	Description
ResultData	Metreos.Types.CiscoIpPhone.CiscoIPPhoneTextType	Represents a minimal CiscoIpPhone.Text message. This value can then be aggregated by a AddSoftKeyItem

Return Value

Returns a properly formatted Cisco IP Phone text screen. The "ResultData" parameter should be assigned to a variable of type "Metreos.Types.CiscoIpPhone.Text".

Remarks

This action creates a simple text display. It utilizes the three main fields on the phone (title, prompt and main text area) to allow basic formatting of the message.

Requirements

Metreos Framework version 1.1

Metreos HTTP Provider version 1.1

Metreos.Native.CiscoIpPhone.SendExecute Action

Sends an execute command to the phone.

Operation

Synchronous

Required Parameters

None

Optional Parameters

Parameter Name	Data Type	Description
Message	System.String	A Cisco IP Phone execute object
Password	System.String	The password for the specified username
URL	System.String	The IP address of a Cisco IP phone.
Username	System.String	A valid username associated with the specified phone

Result Data Fields

Parameter Name	Data Type	Description
ResultData	System.Object	Represents a Cisco IP Phone result message.

Return Value

Returns "success" if the phone was reachable and login was successful. Otherwise, "failure" is returned.

The two most common failure scenarios are if the phone is not reachable at the given URL or if the phone returned an error after receiving the request.

Remarks

This action sends a "Metreos.Types.CiscoIpPhone.Execute" object to a Cisco IP phone by way of an HTTP "PUT" request. The phone will then fetch the resource indicated by the URL in the Execute object.

The response received from the phone is parsed, placed in a "Metreos.Types.CiscoIpPhone.Result" object and passed back to the application.

Requirements

Metreos Framework version 1.1

Metreos HTTP Provider version 1.1

Types

Metreos Framework Package Library

Metreos.Types.CiscoIpPhone.Directory Type

Represents a directory display.

Serializable

Yes

Input Types

Type Name	Description
CiscoIPPhoneDirectoryType	The result of a CiscoIpPhone.CreateDirectory action
CiscoIPPhoneDirectoryEntryType	The result of a CiscoIpPhone.AddDirectoryEntry action
CiscoIPPhoneSoftKeyType	The result of a CiscoIpPhone.AddSoftKeyItem action

Accessible Public Methods

None.

Accessible Public Properties

None.

Remarks

Once initialized and populated, send this type as the body of the HTTP response to the phone's request. Be sure to set the "Content-Type" header of the response to "text/xml". Also, best practice dictates that the HTTP "Expires" header should be set to "-1".

Requirements

Metreos Framework version 1.1

Metreos.Types.CiscoIpPhone.GraphicFileMenu Type

Represents a graphic file menu display.

Serializable

Yes

Input Types

Type Name	Description
CiscoIPPhoneGraphicFileMenuType	The result of a CiscoIpPhone.CreateGraphicFileMenu action
CiscoIPPhoneMenuItemType	The result of a CiscoIpPhone.AddMenuItem action
CiscoIPPhoneSoftKeyType	The result of a CiscoIpPhone.AddSoftKeyItem action

Accessible Public Methods

None.

Accessible Public Properties

None.

Remarks

Once initialized and populated, send this type as the body of the HTTP response to the phone's request. Be sure to set the "Content-Type" header of the response to "text/xml". Also, best practice dictates that the HTTP "Expires" header should be set to "-1".

Requirements

Metreos Framework version 1.1

Metreos.Types.CiscoIpPhone.GraphicMenu Type

Represents a graphic menu display.

Serializable

Yes

Input Types

Type Name	Description
CiscoIPPhoneGraphicMenuType	The result of a CiscoIpPhone.CreateGraphicMenu action
CiscoIPPhoneMenuItemType	The result of a CiscoIpPhone.AddMenuItem action
CiscoIPPhoneSoftKeyType	The result of a CiscoIpPhone.AddSoftKeyItem action

Accessible Public Methods

None.

Accessible Public Properties

None.

Remarks

Once initialized and populated, send this type as the body of the HTTP response to the phone's request. Be sure to set the "Content-Type" header of the response to "text/xml". Also, best practice dictates that the HTTP "Expires" header should be set to "-1".

Requirements

Metreos Framework version 1.1

Metreos.Types.CiscoIpPhone.IconMenu Type

Represents an icon menu display.

Serializable

Yes

Input Types

Type Name	Description
CiscoIPPhoneIconMenuType	The result of a CiscoIpPhone.CreateIconMenu action
CiscoIPPhoneMenuItemType	The result of a CiscoIpPhone.AddMenuItem action
CiscoIPPhoneSoftKeyType	The result of a CiscoIpPhone.AddSoftKeyItem action
CiscoIPPhoneIconItemType	The result of a CiscoIpPhone.AddIconItem action

Accessible Public Methods

None.

Accessible Public Properties

None.

Remarks

Once initialized and populated, send this type as the body of the HTTP response to the phone's request. Be sure to set the "Content-Type" header of the response to "text/xml". Also, best practice dictates that the HTTP "Expires" header should be set to "-1".

Requirements

Metreos Framework version 1.1

Metreos.Types.CiscoIpPhone.Image Type

Represents an image display.

Serializable

Yes

Input Types

Type Name	Description
CiscoIPPhoneImageType	The result of a CiscoIpPhone.CreateImage action
CiscoIPPhoneSoftKeyType	The result of a CiscoIpPhone.AddSoftKeyItem action

Accessible Public Methods

None.

Accessible Public Properties

None.

Remarks

Once initialized and populated, send this type as the body of the HTTP response to the phone's request. Be sure to set the "Content-Type" header of the response to "text/xml". Also, best practice dictates that the HTTP "Expires" header should be set to "-1".

Requirements

Metreos Framework version 1.1

Metreos.Types.CiscoIpPhone.Directory Type

Represents an image file display.

Serializable

Yes

Input Types

Type Name	Description
CiscoIPPhoneImageFileType	The result of a CiscoIpPhone.CreateImage action
CiscoIPPhoneSoftKeyType	The result of a CiscoIpPhone.AddSoftKeyItem action

Accessible Public Methods

None.

Accessible Public Properties

None.

Remarks

Once initialized and populated, send this type as the body of the HTTP response to the phone's request. Be sure to set the "Content-Type" header of the response to "text/xml". Also, best practice dictates that the HTTP "Expires" header should be set to "-1".

Requirements

Metreos Framework version 1.1

Metreos.Types.CiscoIpPhone.Input Type

Represents an input prompt.

Serializable

Yes

Input Types

Type Name	Description
CiscoIPPhoneInputType	The result of a CiscoIpPhone.CreateInput action
CiscoIPPhoneInputItemType	The result of a CiscoIpPhone.AddInputItem action
CiscoIPPhoneSoftKeyType	The result of a CiscoIpPhone.AddSoftKeyItem action

Accessible Public Methods

None.

Accessible Public Properties

None.

Remarks

Once initialized and populated, send this type as the body of the HTTP response to the phone's request. Be sure to set the "Content-Type" header of the response to "text/xml". Also, best practice dictates that the HTTP "Expires" header should be set to "-1".

Requirements

Metreos Framework version 1.1

Metreos.Types.CiscoIpPhone.Menu Type

Represents a simple menu display.

Serializable

Yes

Input Types

Type Name	Description
CiscoIPPhoneMenuType	The result of a CiscoIpPhone.CreateMenu action
CiscoIPPhoneMenuItemType	The result of a CiscoIpPhone.AddMenuItem action
CiscoIPPhoneSoftKeyType	The result of a CiscoIpPhone.AddSoftKeyItem action

Accessible Public Methods

None.

Accessible Public Properties

None.

Remarks

Once initialized and populated, send this type as the body of the HTTP response to the phone's request. Be sure to set the "Content-Type" header of the response to "text/xml". Also, best practice dictates that the HTTP "Expires" header should be set to "-1".

Requirements

Metreos Framework version 1.1

Metreos.Types.CiscoIpPhone.Text Type

Represents a text display.

Serializable

Yes

Input Types

Type Name	Description
CiscoIPPhoneTextType	The result of a CiscoIpPhone.CreateText action
CiscoIPPhoneSoftKeyType	The result of a CiscoIpPhone.AddSoftKeyItem action

Accessible Public Methods

None.

Accessible Public Properties

None.

Remarks

Once initialized and populated, send this type as the body of the HTTP response to the phone's request. Be sure to set the "Content-Type" header of the response to "text/xml". Also, best practice dictates that the HTTP "Expires" header should be set to "-1".

Requirements

Metreos Framework version 1.1

Metreos.Types.CiscoIpPhone.Response Type

Represents the response to a CiscoIpPhone.SendExecute action.

Serializable

Yes

Input Types

Type Name	Description
CiscoIPPhoneResponseType	One possible result of a CiscoIpPhone.SendExecute action
CiscoIPPhoneErrorType	The other possible result of a CiscoIpPhone.SendExecute action

Accessible Public Methods

None.

Accessible Public Properties

Type Name	Description
bool IsError	Returns a boolean indicated whether the phone responded with an error.

Remarks

In this release, this type only identifies if an error occurred, but it does not return the actual error text. It is possible however to forward the response as XML to a client for display by serializing this type in the body of an HTTP "POST" request.

Requirements

Metreos Framework version 1.1

Metreos.Types.CiscoIpPhone.Execute Type

Represents an "execute" command.

Serializable

Yes

Input Types

Type Name	Description
CiscoIPPhoneExecuteType	The result of a CiscoIpPhone.CreateExecute action

Accessible Public Methods

None.

Accessible Public Properties

None.

Remarks

This type differs from the other Cisco IP phone types in that it is sent in the body of an HTTP request instead of a response. Use the "SendExecute" action to send the request to the phone.

Requirements

Metreos Framework version 1.1

8. TIMER FACILITY

Actions

Metreos Framework Package Library

Metreos.Providers.TimerFacility.AddTriggerTimer Action

Add a new low-resolution timer which will trigger a new script instance.

Operation

Synchronous

Required Parameters

Parameter Name	Data Type	Description
timerDateTime	System.DateTime	Initial time to fire. Must be in a .NET Framework System.DateTime parseable format

Optional Parameters

Parameter Name	Data Type	Description
timerRecurrenceInterval	System.TimeSpan	Period of the timer. Must be in a .NET Framework System.TimeSpan parseable format
timerUserData	System.String	An opaque token used to allow distinguishable timer events to be raised

Result Data Fields

Parameter Name	Data Type	Description
timerId	System.String	Unique timer identifier

Return Value

Returns “success” if the timer was added; otherwise, “failure” is returned.

Remarks

Adds a new timer to the timer table of the Timer Facility Provider. This action will fire Metreos.Providers.TimerFacility.TimerFired , which will trigger a new script.

Applications may specify a recurrence interval for the timer. This is a period of time over which the timer will repeat until removed by the application. Further, “timerUserData” will be delivered to the application in the “TimerFire” event.

Requirements

Metreos Framework version 1.1

Metreos Timer Facility Provider version 1.1

Metreos.Providers.TimerFacility.AddNonTriggerTimer Action

Add a new low-resolution timer which will fire an event in the script instance which invoked this action.

Operation

Synchronous

Required Parameters

Parameter Name	Data Type	Description
timerDateTime	System.DateTime	Initial time to fire. Must be in a .NET Framework System.DateTime parseable format

Optional Parameters

Parameter Name	Data Type	Description
timerRecurrenceInterval	System.TimeSpan	Period of the timer. Must be in a .NET Framework System.TimeSpan parseable format
timerUserData	System.String	An opaque token used to allow distinguishable timer events to be raised

Result Data Fields

Parameter Name	Data Type	Description
timerId	System.String	Unique timer identifier

Return Value

Returns “success” if the timer was added; otherwise, “failure” is returned.

Remarks

Adds a new timer to the timer table of the Timer Facility Provider. The timer, when fired, will raise a Metreos.Providers.TimerFacility.TimerFired event in the script instance which called Metreos.Providers.TimerFacility.AddNonTriggerTimer.

Applications may specify a recurrence interval for the timer. This is a period of time over which the timer will repeat until removed by the application. Further, “timerUserData” will be delivered to the application in the “TimerFire” event.

Requirements

Metreos Framework version 1.1

Metreos Timer Facility Provider version 1.1

Metreos.Providers.TimerFacility.RemoveTimer Action

Remove a timer that has been previously added by an application.

Operation

Synchronous

Required Parameters

Parameter Name	Data Type	Description
timerId	System.String	Unique timer identifier

Optional Parameters

None

Result Data Fields

None

Return Value

Returns "success" if the timer was successfully removed; otherwise, "failure" is returned.

Remarks

Timers that have a recurrence interval specified will continue to fire until removed by the application that added them. Further, it is best practice to always remove a timer even if no recurrence interval is specified. The "timerId" parameter indicates which timer should be removed from the Timer Facility Provider's timer table.

Requirements

Metreos Framework version 1.1

Metreos Timer Facility Provider version 1.1

Events

Metreos Framework Package Library

Metreos.Providers.TimerFacility.TimerFire Event

Event fired when a previously added timer is triggered.

Type

Non-Triggering

Event Parameters

Parameter Name	Data Type	Description
timerId	System.String	Unique timer identifier
timerUserData	System.String	An opaque token used to allow distinguishable timer events to be raised

Remarks

The "TimerFire" event is fired by the Timer Facility Provider when a previously added timer is triggered. The event will contain both the ID of the timer that has fired along with the user data specified in the "AddTimer" action.

Requirements

Metreos Framework version 1.1

Metreos Timer Facility Provider version 1.1

9. CISCO DEVICELISTX

The Cisco DeviceListX provider included with the Metreos Communications Environment enables the device list data to be cached locally on the Metreos Application Server. This enables running applications to query the device list data without concern for impacting the performance of their Cisco CallManager.

Actions

Metreos Framework Package Library

Metreos.Providers.CiscoDeviceListX.Refresh Action

The Refresh action causes the DeviceListX provider to initiate a manual refresh of the device list data cache.

Operation

Asynchronous

Required Parameters

Parameter Name	Data Type	Description
userData	System.String	An opaque token used to correlate asynchronous actions with final responses in an application

Optional Parameters

None.

Result Data Fields

None.

Return Value

Returns a provisional response of "success" if the DeviceListX provider was able to initiate a refresh of the device list data cache, otherwise "failure" is returned.

A final response will be returned to the application in the form of an event when the device list data cache refresh has completed. A "Refresh_Complete" event indicates a successful final response. Likewise, a "Refresh_Failed" event indicates that an error occurred and the refresh was unable to complete.

Remarks

When the DeviceListX provider initiates a device list data cache refresh it must contact all of the configured CallManager subscribers and request a new device list data dump. This could potentially take a very long time and applications must be aware of this when using the "Refresh" action.

To reduce the impact to Cisco CallManager the DeviceListX provider will not process back-to-back refresh requests. In this situation those requests that were not processed will be indicated as having failed. Furthermore, if an automatic refresh has been recently completed the manual refresh request may result in a failure.

Requirements

Metreos Framework version 1.1

Metreos Cisco DeviceListX Provider version 1.1

Metreos.Native.CiscoDeviceList.Query Action

Queries the Cisco DeviceListX cache for device information.

Operation

Synchronous

Required Parameters

None.

Optional Parameters

Parameter Name	Data Type	Description
Description	System.String	Description of the device
IP	System.String	IP address of the device
Name	System.String	Name of the device
Pool	System.String	Device pool in which this device is contained
SearchSpace	System.String	Search space of the device
Status	System.String	Status of the device
Type	System.String	Type code of the device

Result Data Fields

Parameter Name	Description
returnValue	System.Data.DataTable

Return Value

Returns "success" if the database query completed without errors; otherwise, "failure". Note that a successful response does not necessarily mean that any devices were found matching the specified criteria.

Remarks

This action will use all the optional parameters supplied to it in an "AND" manner to locate a matching device definition. The resulting DataTable object should be assigned to a "Metreos.Types.CiscoDeviceList.ResultData" variable. Then the rows can be dealt with as individual DataRow objects.

Requirements

Metreos Framework version 1.1

Metreos Cisco DeviceListX Provider version 1.1

Asynchronous Callback Events

Metreos Framework Package Library

Metreos.Providers.CiscoDeviceListX.Refresh_Complete Event

Event fired to the application indicating successful completion of a previously initiated "Refresh" action.

Type

Asynchronous Callback

Event Parameters

Parameter Name	Data Type	Description
userData	System.String	An opaque token used to correlate asynchronous actions with final responses in an application

Remarks

None.

Requirements

Metreos Framework version 1.1

Metreos Cisco DeviceListX Provider version 1.1

Metreos.Providers.CiscoDeviceListX.Refresh_Failed Event

Event fired to the application indicating failure of a previously initiated “Refresh” action.

Type

Asynchronous Callback

Event Parameters

Parameter Name	Data Type	Description
userData	System.String	An opaque token used to correlate asynchronous actions with final responses in an application

Remarks

None.

Requirements

Metreos Framework version 1.1

Cisco DeviceListX Provider version 1.1

10. TAPI

All actions and events related to the TAPI provider must be considered according to the TAPI paradigm. In order to begin performing any actions or receiving any events, the provider must be configured to monitor one or more lines.

If the lines of both parties involved in a call are monitored, there will be two call IDs associated with the call. One “callId” will be returned from the “MakeCall” action and the other will come in as a parameter on the “IncomingCall” event. It is critical to keep these straight since the “callId” parameter implies the line on which the desired action should be taken. Additionally, certain events, such as “CallEstablished”, will be fired once for each monitored line.

Note also that the functionality of all TAPI actions and events may be altered or limited by the particular TAPI service provider (TSP) in use. Since not all TAPI providers implement all TAPI methods, consult the documentation for your TSP before invoking actions on the Metreos TAPI provider.

Actions

Metreos Framework Package Library

Metreos.Providers.Tapi.Accept Action

Accepts, but does not answer, an incoming 3rd party call on a monitored line.

Operation

Synchronous

Required Parameters

Parameter Name	Data Type	Description
callId	System.String	A unique token used to identify this particular call leg in future actions and events

Optional Parameters

None

Result Data Fields

None

Return Value

Returns “success” if all basic conditions were met to accept the call, otherwise “failure”.

Remarks

Invoking this action affirms to the calling party that the called party is present on the network, but does not affirm that they wish to answer the call. The “callId” field should match the value received in the “IncomingCall” event.

For more information, see the TAPI specification.

Requirements

Metreos Framework version 1.1

Metreos TAPI provider version 1.1

Metreos.Providers.Tapi.AnswerCall Action

Performs a 3rd party answer of a pending call on a monitored TAPI line

Operation

Asynchronous

Required Parameters

Parameter Name	Data Type	Description
answer	System.Boolean	A boolean indicating whether the call should be answered or rejected
callId	System.String	A unique token used to identify this particular call leg in future actions and events
userData	System.String	An opaque token used to correlate asynchronous actions with final responses in an application

Optional Parameters

None

Result Data Fields

Parameter Name	Data Type	Description
callId	System.String	A unique token used to identify this particular call leg in future actions and events

Return Value

Returns "success" if all basic conditions were met to begin answering the call, otherwise "failure".

Remarks

This action will answer a call where the called party is on a monitored line. The "callId" parameter should be set to the value of the same parameter in the "IncomingCall" event.

Once the call has been successfully answered, an "AnswerCall_Complete" event will be fired to the application. If an error occurred while answering the call, an "AnswerCall_Failed" event will be fired to the application.

Requirements

Metreos Framework version 1.1

Metreos TAPI provider version 1.1

Metreos.Providers.Tapi.BlindTransfer Action

Forwards an established 3rd party phone call to another destination

Operation

Synchronous

Required Parameters

Parameter Name	Data Type	Description
callId	System.String	A unique token used to identify this particular call leg in future actions and events
to	System.String	The dialed number of the destination endpoint to transfer the incoming call to

Optional Parameters

None

Result Data Fields

None

Return Value

Returns "success" if all basic conditions were met to transfer the call, otherwise "failure".

Remarks

This action can be invoked at any time on an active call to forward it to another party. The "callId" parameter implies the call and line which is to be forwarded. If the "to" parameter specifies an invalid DN, a "failure" response will be returned. Otherwise, it is assumed that the transfer was completed successfully.

This action should only be used on calls which have been answered. If you wish to forward a call that has not yet been answered, use "Redirect" instead.

In future implementations, this will become an asynchronous action where the final response will indicate whether the destination party actually answered the transferred call successfully.

Requirements

Metreos Framework version 1.1

Metreos TAPI provider version 1.1

Metreos.Providers.Tapi.Hangup Action

Terminates an established 3rd party phone call.

Operation

Asynchronous

Required Parameters

Parameter Name	Data Type	Description
callId	System.String	A unique token used to identify this particular call leg in future actions and events
userData	System.String	An opaque token used to correlate asynchronous actions with final responses in an application

Optional Parameters

None

Result Data Fields

None

Return Value

Returns “success” if all basic conditions were met to begin terminating the call, otherwise “failure”.

Remarks

This action terminates a call on a monitored line. It doesn’t matter if the line is the calling or called party.

Requirements

Metreos Framework version 1.1

Metreos TAPI provider version 1.1

Metreos.Providers.Tapi.MakeCall Action

Initiates a 3rd party call between two endpoints.

Operation

Asynchronous

Required Parameters

Parameter Name	Data Type	Description
from	System.String	The number of the originating endpoint
to	System.String	The number of the destination endpoint
userData	System.String	An opaque token used to correlate asynchronous actions with final responses in an application

Optional Parameters

None

Result Data Fields

Parameter Name	Data Type	Description
callId	System.String	A unique token used to identify this particular call leg in future actions and events

Return Value

Returns "success" if all conditions were met to initiate the call (e.g. line exists, source phone was on hook, etc); otherwise, "failure" is returned.

Remarks

The provider must be configured to monitor one or more lines before this action will complete successfully. The "from" field should designate the DN of the source device which must be registered to a monitored line. The "to" field can be any valid DN reachable by the communications system.

The "callId" returned in the result is the unique identifier for the call and is used to correlate all subsequent actions and events related to this call.

Once the call has been completed successfully a "MakeCall_Complete" event will be fired to the application. If an error occurs while making the call, a "MakeCall_Failed" event will be fired to the application.

Requirements

Metreos Framework version 1.1

Metreos TAPI provider version 1.1

Metreos.Providers.Tapi.Redirect Action

Redirects an offered call to a specified party

Operation

Synchronous

Required Parameters

Parameter Name	Data Type	Description
callId	System.String	A unique token used to identify this particular call leg in future actions and events
to	System.String	The number of the destination endpoint to transfer the incoming call

Optional Parameters

None

Result Data Fields

None

Return Value

Returns “success” if all basic conditions were met to redirect the call, otherwise “failure”.

Remarks

This action can be invoked on an offered call to redirect it to a new destination. This action should not be executed on answered calls. If you wish to forward a call which has already been answered, use “BlindTransfer” instead.

Requirements

Metreos Framework version 1.1

Metreos TAPI provider version 1.1

Events

Metreos Framework Package Library

Metreos.Providers.Tapi.CallEstablished Event

Event fired indicating that a call has been established.

Type

Triggering

Event Parameters

Parameter Name	Data Type	Description
callId	System.String	A unique token used to identify this particular call leg in future actions and events

Remarks

This event will come in anytime a monitored line successfully establishes a call. If both parties in the call are being monitored, this event will be fired twice, once for each line. The "callId" returned will be different since the "callId" implies both the call and the line.

Requirements

Metreos Framework version 1.1

Metreos TAPI provider version 1.1

Metreos.Providers.Tapi.Hangup Event

Event fired indicating that a call on a monitored line has been terminated

Type

Triggering

Event Parameters

Parameter Name	Data Types	Description
callId	System.String	A unique token used to identify this particular call leg in future actions and events

Remarks

This event will come in anytime a call on a monitored line is terminated. If both parties in the call are being monitored, this event will be fired twice, once for each line. The "callId" returned will be different since the "callId" implies both the call and the line.

Requirements

Metreos Framework version 1.1

Metreos TAPI provider version 1.1

Metreos.Providers.Tapi.IncomingCall Event

Event fired indicating that a monitored line is being offered a call.

Type

Triggering

Event Parameters

Parameter Name	Data Type	Description
callId	System.String	A unique token used to identify this particular call leg in future actions and events
from	System.String	The number of the originating endpoint
to	System.String	The number of the destination endpoint

Remarks

This event will provide the necessary details to take action on a call being offered to a monitored line. "AnswerCall" should always be executed in response to this event. If you wish to answer the call, set the "answer" parameter on the "AnswerCall" action to "true". Otherwise, set it to "false".

Requirements

Metreos Framework version 1.1

Metreos TAPI provider version 1.1

Asynchronous Callback Events

Metreos Framework Package Library

Metreos.Providers.Tapi.AnswerCall_Complete Event

Event fired to indicate that the "AnswerCall" action completed successfully.

Type

Asynchronous Callback

Event Parameters

Parameter Name	Data Type	Description
callId	System.String	A unique token used to identify this particular call leg in future actions and events
userData	System.String	An opaque token used to correlate asynchronous actions with final responses in an application

Remarks

This is returned when the TAPI service provider indicates that the incoming call has been answered successfully.

Requirements

Metreos Framework version 1.1

Metreos TAPI provider version 1.1

Metreos.Providers.Tapi.AnswerCall_Failed Event

Event fired to indicate that the "AnswerCall" action completed unsuccessfully.

Type

Asynchronous Callback

Event Parameters

Parameter Name	Data Type	Description
callId	System.String	A unique token used to identify this particular call leg in future actions and events
userData	System.String	An opaque token used to correlate asynchronous actions with final responses in an application

Remarks

This callback indicates that the TAPI service provider was unable to answer the call.

Requirements

Metreos Framework version 1.1

Metreos TAPI provider version 1.1

Metreos.Providers.Tapi.MakeCall_Complete Event

Event fired indicating that the server-initiated 3rd party phone call was established successfully.

Type

Asynchronous Callback

Event Parameters

Parameter Name	Data Type	Description
callId	System.String	A unique token used to identify this particular call leg in future actions and events
userData	System.String	An opaque token used to correlate asynchronous actions with final responses in an application

Remarks

This callback indicates to the application that the "MakeCall" action has completed successfully. This means that the remote party has accepted and answered the offered call.

Requirements

Metreos Framework version 1.1

Metreos TAPI provider version 1.1

Metreos.Providers.Tapi.MakeCall_Failed Event

Event fired indicating that the server-initiated 3rd party phone call was not established.

Type

Asynchronous Callback

Event Parameters

Parameter Name	Data Type	Description
callId	System.String	A unique token used to identify this particular call leg in future actions and events
userData	System.String	An opaque token used to correlate asynchronous actions with final responses in an application

Remarks

This callback indicates to the application that the "MakeCall" action did not complete successfully. This means that the remote party either rejected the call or a ring out condition was reached.

Requirements

Metreos Framework version 1.1

Metreos TAPI provider version 1.1

11. DIAL PLAN

Actions

Metreos Framework Package Library

Metreos.Native.DialPlan.FormatAddress Action

This action applies the configured dial plan to the specified dialed number (DN).

Operation

Synchronous

Required Parameters

Parameter Name	Data Type	Description
dialedNumber	System.String	The number to format. Can be already fully qualified with the communications system's IP address

Optional Parameters

Parameter Name	Data Type	Description
cmAddress	System.String	IP address of outbound gateway
dialingRules	System.Collections.Hashtable	A hash table containing patterns to apply to the fully qualified number, with replacement strings to apply to a matching substring, if one occurs. Keys are the patterns, values are the replacements

Result Data Fields

Parameter Name	Data Type	Description
returnValue	System.String	The formatted address with all the dialing rules applied

Return Value

Returns "success" if enough parameters where specified to successfully format the DN. Otherwise, "failure" is returned.

Remarks

This action will format a given DN into the "dialedNumber@serverAddress" notation. If the DN is already in that notation, the "cmAddress" parameter may be omitted. Otherwise, the "cmAddress" parameter must be present so that it can be suffixed to the "dialedNumber@" portion of the address.

The "dialingRules" parameter is a hash table of regular expressions which represent the dialing rule to be applied. The hash table key is the pattern to match and the value is the regular expression transformation to apply.

Requirements

Metreos Framework version 1.1

12. MAIL

Actions in this namespace support sending SMTP email. Supported email functionality includes sending as HTML, one or more attachments, and SMTP authentication.

Actions

Metreos Framework Package Library

Metreos.Native.Mail.Send Action

Send an SMTP formatted email.

Operation

Synchronous

Required Parameters

Parameter Name	Data Type	Description
to	System.String	A recipient for the email

Optional Parameters

Parameter Name	Data Type	Description
attachmentPath	System.String	A StringCollection of files to attach to the email
authenticationMode	System.String	Authentication mode mandated by mail server (default: Base64)
body	System.String	The body of the email
from	System.String	Identity to use as sender
mailServer	System.String	Mail server to use if different than the domain of the recipient
password	System.String	Mail server credentials
sendAsHtml	System.Boolean	Boolean indicating whether to send as HTML (default: false)
subject	System.String	The subject of the email
username	System.String	Mail server credentials

Result Data Fields

None.

Return Value

Returns “success” if the mail was sent successfully, otherwise “failure”.

Remarks

This action sends an email to the specified recipient using the specified server. All basic SMTP functions are supported including authentication and attachments. The MIME type of the attachments is auto-detected.

The action must include “username” and “mailServer” or “from” or both.

The action must also include “subject” or “body” or “attachPaths” or any combination of these.

Requirements

Metreos Framework version 1.1

13. STANDARD TYPES

Following is a list of types which should for all purposes be treated the same as their .NET counterpart:

Type Name
Metreos.Types.ArrayList
Metreos.Types.Bool
Metreos.Types.DataSet
Metreos.Types.DataTable
Metreos.Types.DateTime
Metreos.Types.Double
Metreos.Types.Hashtable
Metreos.Types.Int
Metreos.Types.Long
Metreos.Types.Queue
Metreos.Types.Short
Metreos.Types.SortedList
Metreos.Types.Stack
Metreos.Types.String
Metreos.Types.StringCollection
Metreos.Types.StringDictionary
Metreos.Types.UInt
Metreos.Types.ULong
Metreos.Types.UShort