



Pexip Infinity

Client REST API v2

Software Version 34

Document Version 34.a

March 2024

]pexip[

Contents

Introduction	2
Using the API	2
Summary of API requests and events	3
Client control requests	7
Conference control functions	11
Participant functions	20
Breakout rooms functions	26
Call functions	28
Server-sent events	31
Other miscellaneous requests	39
Changelog	40
More information	41

Introduction

This guide describes the Pexip client REST API. It is designed for use by non-web-based, third-party voice/video applications that want to initiate or connect to conferences hosted on the Pexip Infinity platform.

We strongly recommend that web-based applications use the [PexRTC JavaScript client API](#) instead.

i This API specification is regularly evolving between versions of the Pexip Infinity platform. While we will attempt to maintain backward compatibility, there may be significant changes between versions.

Using the API

The prefix for all conference-related API calls is:

`https://<node_address>/api/client/v2/conferences/<conference_alias>/`

where `<node_address>` is the address of a Conferencing Node and `<conference_alias>` is an alias of the conference you are connecting to. Under this API path comes a sequence of response API calls, for example:

`https://10.0.0.1/api/client/v2/conferences/meet_alice/request_token`

All commands in the client API are authenticated with a token, which is presented by the Pexip Conferencing Node. The token has a validity lifetime, before the end of which it must be refreshed. The token is presented in a HTTP header entitled "token" on every HTTP request, except for the initial `request_token` request.

Unless otherwise specified, all payloads of requests and responses are JSON objects, Content-Type: application/json.

The responses have two fields, `status` and `result`:

- `status` is "success" if the command has been processed by Pexip, or "failure" if the command could not be processed. Note that this does not mean that the end result is success, only that the request has been received and processed.
- the `result` field indicates if the request was successful.

Summary of API requests and events

This section summarizes the requests and server-sent events that may be used, which are then described in more detail.

Client control requests summary

These REST URIs take the format:

`https://<node_address>/api/client/v2/conferences/<conference_alias>/<request>`

Request	GET/POST	Description
request_token	POST	Requests a new token from the Pexip Conferencing Node.
refresh_token	POST	Refreshes a token to get a new one.
release_token	POST	Releases the token (effectively a disconnect for the participant).

Conference control functions summary

These REST URIs take the format:

`https://<node_address>/api/client/v2/conferences/<conference_alias>/<request>`

Request	GET/POST	Description
dial	POST	Dials out from the conference to a target endpoint.
conference_status	GET	Provides the status of the conference.
lock / unlock	POST	Locks / unlocks the conference.
start_conference	POST	Starts a conference and allows Guests in the "waiting room" to join the meeting.
muteguests / unmuteguests	POST	Mutes / unmutes all Guests on a conference.
disconnect	POST	Disconnects all conference participants, including the participant calling the function.
message	POST	Sends a message to all participants in the conference.
participants	GET	Returns the full participant list of the conference.
transform_layout	POST	Changes the conference layout, controls streaming content, and enables/disables indicators and overlay text.
clearallbuzz	POST	Lower all raised hands.
silent_video_detection	POST	Configure the parameters for silent video detection in an Adaptive Composition layout.
set_classification_level	POST	Sets the classification level to use from the theme assigned to the conference.
get_classification_level	GET	Gets the conference's classification levels and current setting.
theme	GET	Provides the theme resources of the conference (direct media only).
available_layouts	GET	Provides a list of all available layouts for the given conference.
layout_svgs	GET	Provides a list of SVG string representations of the layouts that are active on the given conference.
breakouts	POST	Creates a breakout room

Participant functions summary

These participant REST URIs take the format:

https://<node_address>/api/client/v2/conferences/<conference_alias>/participants/<participant_uuid>/<request>

Request	GET/POST	Description
disconnect	POST	Disconnects a participant.
mute / unmute	POST	Mutes / unmutes a participant's audio.
video_muted / video_unmuted	POST	Mutes / unmutes a participant's video.
allowrxpresentation / denyrxpresentation	POST	Enables or disables a participant from receiving the presentation stream.
spotlighton / spotlightoff	POST	Enables or disables the "spotlight" on a participant.
unlock	POST	Lets a specified participant into the conference from the waiting room of a locked conference.
dtmf	POST	Sends DTMF digits to the participant.
message	POST	Sends a message to the participant.
calls	POST	Upgrades this connection to have a WebRTC audio / video / presentation call element.
overlaytext	POST	Changes the participant name overlay text.
pres_in_mix	POST	Controls whether or not the participant sees presentation in the layout mix (Adaptive Composition layout only).
role	POST	Changes the role of the participant.
fecc	POST	Send Far End Camera Control messaging to the participant.
buzz	POST	Raise a participant's hand.
clearbuzz	POST	Lower a participant's hand.
transfer	POST	Transfers a participant to another conference.
take_floor	POST	Signals that the participant is starting sending full-motion presentation.
release_floor	POST	Signals that the participant has finished sending full-motion presentation.
avatar.jpg	GET	Obtains the image to display to represent a conference participant or directory contact.
statistics	POST	Report participant media statistics to Pexip Infinity (direct media only).
preferred_aspect_ratio	POST	Specifies the aspect ratio the participant would like to receive.

Breakout room functions summary

Many breakout room REST URIs take the format:

https://<node_address>/api/client/v2/conferences/<conference_alias>/breakouts/<breakout_uuid>/<request>

See [Breakout rooms functions](#) for details.

Call functions summary

These call REST URIs take the format:

`https://<node_address>/api/client/v2/conferences/<conference_alias>/participants/<participant_uuid>/calls/<call_uuid>/<request>`

Request	GET/POST	Description
<u>ack</u>	POST	Starts media for the specified call (WebRTC calls only).
<u>disconnect</u>	POST	Disconnects the specified call.
<u>dtmf</u>	POST	Sends DTMF digits to the specified participant.
<u>new_candidate</u>	POST	Send a new ICE candidate if doing trickle ICE.
<u>update</u>	POST	Sends a new SDP.
<u>fecc</u>	POST	Sends Far End Camera Control messaging to the participant.

Server-sent events summary

To subscribe, open an HTTP connection to:

`https://<node_address>/api/client/v2/conferences/<conference_alias>/events?token=<token_id>`

Event	Description
<u>presentation_start</u>	Marks the start of a presentation, and includes the information on which participant is presenting.
<u>presentation_stop</u>	The presentation has finished.
<u>presentation_frame</u>	A new presentation frame is available.
<u>participant_create</u>	A new participant has joined the conference.
<u>participant_update</u>	A participant's properties have changed.
<u>participant_delete</u>	A participant has left the conference.
<u>participant_sync_begin</u> / <u>participant_sync_end</u>	These two messages start and end the sending of the complete participant list.
<u>conference_update</u>	Conference properties have been updated.
<u>layout</u>	The stage layout has changed.
<u>message_received</u>	A chat message has been broadcast to the conference.
<u>stage</u>	An update to the "stage layout" is available. This declares the order of active speakers, and their voice activity.
<u>call_disconnected</u>	Sent when a child call has been disconnected.
<u>disconnect</u>	Sent when the participant is being disconnected from the Pexip side.
<u>splash_screen</u>	The client should display a splash screen,
<u>new_offer</u>	Received a SDP offer from the far-end participant (direct media only).
<u>update_sdp</u>	Received a SDP update from the far-end participant (direct media only).
<u>new_candidate</u>	Received a new ICE candidate from the far-end participant (direct media only).
<u>peer_disconnect</u>	Indicates that the far-end participant has disconnected (direct media only).

Event	Description
refer	The client should connect to a different conference.
breakout_begin	A breakout room has been created.
breakout_event	Breakout room events that are occurring within the breakout VMR.
breakout_end	A breakout room has been closed.
breakout_refer	The client should connect to a different breakout room.

Other requests summary

Request	Description
/api/client/v2/status	Check whether a Conferencing Node is in maintenance mode.

Client control requests

This section describes in detail the requests that may be used to initiate and manage a connection to a Conferencing Node.

These REST URIs take the format:

`https://<node_address>/api/client/v2/conferences/<conference_alias>/<request>`

request_token

This POST requests a new token from the Pexip Conferencing Node.

Request example:

```
{"display_name": "Alice", "call_tag": "def456"}
```

Request fields:

display_name	string	The name by which this participant should be known.
call_tag	string	An optional call tag to assign to this participant.
direct_media	boolean	Indicates to Pexip Infinity if the client has support for direct media.

Response example:

```
{'status': 'success',
 'result': {'token': 'Ohv3IrcV0CJFa ... etc ... de2PvYam0zqTlG8fEA==',
 'expires': '120',
 'display_name': 'pmxep',
 'participant_uuid': '4cb84ae2-3359-4aa6-9e16-b6fa43f8af3a',
 'current_service_type': 'conference',
 'role': 'HOST',
 'call_tag': '',
 'version': {'version_id': '29', 'pseudo_version': '68488.0.0'},
 'conference_name': 'Alice's VMR',
 'chat_enabled': True,
 'fecc_enabled': True,
 'rtmp_enabled': True,
 'analytics_enabled': True,
 'service_type': 'conference',
 'call_type': 'video',
 'guests_can_present': True,
 'vp9_enabled': False,
 'allow_1080p': True,
 'direct_media': False,
 'use_relay_candidates_only': False,
 'turn': [{'urls': ['turn:176.125.235.148:3478?transport=udp'],
 'username': '1660056163:abcdefgh-3359-4aa6-9e16-b6fa43f8af3a',
 'credential': '0BKTlabcdefghvpYMroo8008+g='}]
 }}
```

This result contains the token (abridged in the above example) to use to authenticate all future requests, and an expiry time (in seconds) after which this token becomes invalid. The full list of fields in the result is as follows:

token	string	The authentication token for future requests.
expires	string	Validity lifetime in seconds. Use refresh_token to obtain an updated token.
allow_1080p	boolean	true = 1080p is enabled; false = 1080p is not enabled.
analytics_enabled	boolean	Whether the Automatically send deployment and usage statistics to Pexip global setting has been enabled on the Pexip installation.

call_tag	string	Optional call tag.
call_type	string	The type of call: <ul style="list-style-type: none">• "video": main video plus presentation• "video-only": main video only• "audio": audio-only
chat_enabled	boolean	true = chat is enabled; false = chat is not enabled.
conference_name	string	The conference name.
current_service_type	string	The service type this user is connecting into. May be "conference", "gateway" or "test_call" as for <code>service_type</code> if directly connecting in. May also be "waiting_room" if waiting to be allowed to join a locked conference, or "ivr" if on the PIN entry screen.
display_name	string	Echoes the display name in the request.
fecc_enabled	boolean	true = FECC is enabled; false = FECC is not enabled.
guests_can_present	boolean	true = guests can present; false = guests cannot present.
idp	array	The response may contain a list of Identity Providers that are available for this service. See SSO protected conferences below for more information.
participant_uuid	string	The uuid associated with this newly created participant. It is used to identify this participant in the participant list.
role	string	Whether the participant is connecting as a "HOST" or a "GUEST".
rtmp_enabled	boolean	true = RTMP is enabled; false = RTMP is not enabled.
service_type	string	Either "conference", "gateway" or "test_call" depending on whether this is a VMR, gateway or Test Call Service respectively.
stun	array	The Client STUN server addresses that are associated with the Conferencing Node's system location.
version	object	The version of the Pexip server being communicated with.
vp9_enabled	boolean	true = vp9 is enabled; false = vp9 is not enabled.
turn	array	The Client TURN server addresses and credentials that are associated with the Conferencing Node's system location.
direct_media	boolean	Specifies if the conference shall use direct media. true = direct media call; false = transcoded call.
pex_datachannel_id	string	ID number (between 0 and 65,534) which uniquely identifies the WebRTC Data Channel (direct media only).
client_stats_update_interval	string	Periodic interval that the client is expected to send media statistics to Pexip Infinity (direct media only).
use_relay_candidates_only	boolean	Whether to enforce media relay on direct media calls. true = direct media call enforcing media relay; this forces only ICE candidates whose IP addresses are being relayed, such as those being passed through a STUN or TURN server, to be considered. See this article for more information. false = direct media call not enforcing media relay.
redirect_url	string	The chosen Identity Provider's URL to redirect to.
redirect_idp	object	The chosen Identity Provider's details.

PIN protected conferences

If the conference is PIN-protected, the PIN must be specified in a "pin" HTTP header. If the PIN is required but is incorrect or missing, a "403 Forbidden" error is returned. The "pin" field in the response specifies whether a PIN is required for Hosts, and a "guest_pin" field in the response specifies whether a PIN is required for Guests. If a PIN is required for a Host, but not for a Guest, and if you want to join as a Guest, you must still provide a "pin" header, with a value of "none".

SSO protected conferences

If the conference has SSO (Single Sign-On) participant authentication enabled, the response contains a list of Identity Providers (IDP) that are available for this service, for example:

```
"idp": [
  {
    "name": "Microsoft Entra ID",
    "img": "",
    "uuid": "4fe19459-67b4-4e84-abb4-a3765a0a7e09"
  }, {
    "name": "ADFS",
    "img": "",
    "uuid": "7e43610f-1c82-4726-899a-af801748846c"
  }
]
```

When using IDP authentication, the next `request_token` request should contain the `chosen_idp`, for example:

```
"chosen_idp": "4fe19459-67b4-4e84-abb4-a3765a0a7e09",
```

And the response will contain a URL to redirect to:

```
"redirect_url": "<IdP's SSO URL>?SAMLRequest=<Base64 encoded SAML AuthNRequest message>",
"redirect_idp": {
  "name": "Microsoft Entra ID",
  "uuid": "4fe19459-67b4-4e84-abb4-a3765a0a7e09"
}
```

After you have received an SSO token from the `redirect_url`, this must be inserted into a third `request_token` request as `sso_token`, along with the `chosen_idp` parameter.

Virtual Receptions

If the conference is a Virtual Reception, a "403 Forbidden" error is returned, with a "conference_extension" field. This field is either:

- "standard": for a regular, Microsoft Teams or Google Meet Virtual Reception.
- "mssip": for a Lync / Skype for Business Virtual Reception.

To join the target room, a second `request_token` request must be made, but with a `conference_extension` field in the request JSON, which contains the alias of the target conference.

refresh_token

This POST request refreshes a token to get a new one.

Request: empty.

Example response:

```
{"status": "success", "result":
{"token": "SE9TVAltZ...etc...jQ4YTVmMzM3MDMwNDF1NjI%3D",
"expires": "120"}}
```

Fields are:

token	string	The new authentication token for future requests.
expires	string	Validity lifetime in seconds.

release_token

This POST request releases the token (effectively a disconnect for the participant).

Request: empty.

Response: should be ignored.

Conference control functions

This section describes in detail the requests that may be used to manage an existing conference.

These REST URIs take the format:

`https://<node_address>/api/client/v2/conferences/<conference_alias>/<request>`

dial

This POST request dials out from the conference to a target endpoint. This function is only available to conference Hosts.

Request example:

```
{"role": "GUEST", "destination": "bob@example.com", "protocol": "sip", "source_display_name": "Alice"}
```

Request fields:

role	string	The level of privileges the participant has in the conference: <ul style="list-style-type: none">"HOST": the participant has Host privileges"GUEST": the participant has Guest privileges
destination	string	The target address to call.
protocol	string	The protocol to use to place the outgoing call: <ul style="list-style-type: none">"sip""h323""rtmp""mssip" (for calls to Microsoft Skype for Business / Lync)"auto" (to use Call Routing Rules) To successfully place calls via the 'auto' protocol option, suitable Call Routing Rules must be configured. To enable calls to be placed via the other protocols you must select Enable legacy dialout API (via Platform > Global Settings > Connectivity).
presentation_url	string	This additional parameter can be specified for RTMP calls to send the presentation stream to a separate RTMP destination.
streaming	string	Identifies the dialed participant as a streaming or recording device: <ul style="list-style-type: none">"yes": streaming/recording participant"no": not a streaming/recording participant Default: "no"
dtmf_sequence	string	An optional DTMF sequence to transmit after the call to the dialed participant starts.
source_display_name	string	Optional field that specifies what the calling display name should be.
source	string	Optional field that specifies the source URI (must be a valid URI for the conference).
call_type	string	Optional field that limits the media content of the call: <ul style="list-style-type: none">"video": main video plus presentation"video-only": main video only"audio": audio-only Default: "video"

keep_conference_alive	string	<p>Determines whether the conference continues when all other non-ADP participants have disconnected:</p> <ul style="list-style-type: none">"keep_conference_alive": the conference continues to run until this participant disconnects (applies to Hosts only)."keep_conference_alive_if_multiple": the conference continues to run as long as there are two or more "keep_conference_alive_if_multiple" participants and at least one of them is a Host."keep_conference_alive_never": the conference terminates automatically if this is the only remaining participant. <p>Default: "keep_conference_alive" for non-streaming participants, and "keep_conference_alive_never" for streaming participants.</p>
remote_display_name	string	An optional friendly name for this participant. This may be used instead of the participant's alias in participant lists and as a text overlay in some layout configurations.
text	string	Optional text to use instead of <code>remote_display_name</code> as the participant name overlay text.

Response example:

```
{"status": "success", "result": ["977fcd1c-8e3c-4dcf-af45-e536b77af088"]}
```

The response is an array of UUIDs of new participants, if dial-out was successfully initiated. In most cases the dial-out will only generate a single call and thus a single UUID in this array, however if Pexip Infinity forks the call there may end up being multiple UUIDs. Only one of these will be answered, however, and the rest will be disconnected.

The call UUIDs will appear as new participants immediately, with a "service_type" of "connecting". If the call is answered, the participant will be updated with a new "service_type", typically being "conference". The participant may also be deleted if the receiver rejects the call, or the call attempt times out in 30 seconds if not answered.

conference_status

This GET request provides the status of the conference.

Currently, a limited set of conference properties are available, as listed below:

```
{"status": "success", "result": {"locked": false, "guests_muted": false, "all_muted": false, "presentation_allowed": true, "started": true, "livecaptions_available": true, "direct_media": false, "breakout": true, "breakout_name": "My Breakout Room", "breakout_description": "Quick discussion about XYZ", "end_action": "transfer", "end_time": 1688688005, "guests_allowed_to_leave": false}}
```

lock / unlock

These POST requests are used to lock or unlock the conference. When a conference is locked, participants waiting to join are held at a "Waiting for Host" screen. These settings are only available to conference Hosts.

Request: empty.

Response: the result is true if successful, false otherwise.

start_conference

If the only user with Host rights is connected to the conference without media (as a presentation and control-only participant), Guests will remain in the "Waiting for Host" screen. This POST request starts the conference and any Guests in the "waiting room" will join the meeting. This is only available to conference Hosts.

Request: empty.

Response: the result is true if successful, false otherwise.

muteguests / unmuteguests

These POST requests are used to mute or unmute all Guests on a conference. When muted, no Guest participants can speak unless they are explicitly unmuted. When unmuted, all Guests on a conference can speak. These settings are only available to conference Hosts.

Request: empty.

Response: the result is true if successful, false otherwise.

disconnect

This POST request disconnects all conference participants, including the participant calling the function. This setting is only available to conference Hosts.

Request: empty.

Response: the result is true if successful, false otherwise.

message

This POST request sends a message to all participants in the conference.

Request example:

```
{"type": "text/plain", "payload": "Hello World"}
```

Request fields:

type	string	The MIME Content-Type. This must be "text/plain" or "application/json".
payload	string	The contents of the message.

Response: the result is true if successful, false otherwise.

participants

This GET request returns the full participant list of the conference. See the description of the [participant_create](#) EventSource for more information.

transform_layout

This POST request changes the conference layout, controls streaming content, and enables/disables indicators and overlay text.

Request fields:

transforms	This is an object containing any of the following optional parameters:	
layout	string	In VMRs the layout for Hosts and Guests is controlled by the <code>layout</code> parameter.
host_layout		In Virtual Auditoriums the Host layout is controlled by the <code>host_layout</code> parameter and the Guest layout is controlled by the <code>guest_layout</code> parameter.
guest_layout		<p>The layout options are:</p> <ul style="list-style-type: none"> "1:0": main speaker only "1:7": main speaker and up to 7 previous speakers "1:21": main speaker and up to 21 previous speakers "2:21": 2 main speakers and up to 21 previous speakers "1:33": 1 small main speaker and up to 33 other speakers "4:0": 2x2 layout, up to a maximum of 4 speakers "9:0": 3x3 layout, up to a maximum of 9 speakers "16:0": 4x4 layout, up to a maximum of 16 speakers "25:0": 5x5 layout, up to a maximum of 25 speakers "one_main_nine_around": one main speaker and up to 9 other participants "one_main_twelve_around": large main speaker and up to 12 other participants "two_mains_eight_around": two main speakers and up to 8 other participants "ac": Adaptive Composition layout "teams": Teams-like layout (this is only suitable for use with Microsoft Teams gateway calls) <p>Note that the <code>layout</code> parameter is an alias for <code>host_layout</code>, and that an attempt to set <code>guest_layout</code> in a service that is not a Virtual Auditorium will return a "400 Bad Request" error.</p>
enable_extended_ac *	boolean	<p>This enables an extended Adaptive Composition (AC) layout that can contain more video participants than the standard AC layout.</p> <p>In the standard AC layout, a maximum of 12 video participants are shown across up to three rows (2 participants on the first row / 3 on the second row / 7 on the bottom row).</p> <p>In the extended layout up to 23 video participants may be shown, initially across three rows (2/3/7 extending to 2/5/7 and then 3/5/7) and then across four rows (3/5/7/8) when required.</p> <p>Note that this setting only has an effect in a conference that is already using AC, so the conference either needs to be already configured to use AC, or you also need to pass <code>"layout": "ac"</code> OR <code>"host_layout": "ac"</code> to enable AC simultaneously, for example:</p> <pre>{"transforms": {"layout": "ac", "enable_extended_ac": true}}</pre> <p>* Technology preview only</p>
streaming_indicator	boolean	Determines whether the streaming indicator icon is disabled (false) or enabled (true).
recording_indicator	boolean	Determines whether the recording indicator icon is disabled (false) or enabled (true).
transcribing_indicator	boolean	Determines whether the transcribing indicator icon is disabled (false) or enabled (true).
enable_active_speaker_indication	boolean	Determines whether active speaker indication is disabled (false) or enabled (true).
enable_overlay_text	boolean	Determines whether participant names overlay text is disabled (false) or enabled (true).
plus_n_pip_enabled	boolean	Controls whether the "plus n indicator" i.e. the number of participants present in the conference, is displayed (true) or not (false).
streaming	object	<p>This can be used to specifically control the content sent to a streaming participant, for example to send a different layout to the stream from that which is seen by standard participants.</p> <p>It is an object containing any of the following optional parameters:</p>

Response: the result is true if successful, false otherwise.

Here are some example requests:

- To change a Virtual Meeting Room layout: {"transforms": {"layout": "2:21"}}
- To change a Virtual Auditorium layout: {"transforms": {"guest_layout": "2:21", "host_layout": "4:0"}}
- To enable streaming and recording indicators: {"transforms": {"streaming_indicator": true, "recording_indicator": true}}
- To enable overlay text: {"transforms": {"enable_overlay_text": true}}
- To set adaptive composition layout with active speaker indication only: {"transforms": {"layout": "ac", "enable_overlay_text": false, "enable_active_speaker_indication": true}}
- To set adaptive composition layout with active speaker indication and to also display all other participant names: {"transforms": {"layout": "ac", "enable_overlay_text": true, "enable_active_speaker_indication": true}}
- To set custom overlay text for a 4:0 layout: {"transforms": {"layout": "4:0", "free_form_overlay_text": ["Top left", "top right", "bottom left", "bottom right"]}}
- To set 2:21 layout for normal participants, and 1:0 for streaming participants: {"transforms": {"layout": "2:21", "streaming": {"layout": "1:0"}}
- To enable the holding screen for streaming participants: {"transforms": {"streaming": {"waiting_screen_enabled": true}}}

clearallbuzz

This POST request lowers all raised hands.

Request: empty.

Response: the result is true if successful, false otherwise.

silent_video_detection

This POST request configures the parameters for silent video detection in an Adaptive Composition layout.

Request fields:

config	This is an object containing any of the following optional parameters:				
	Name	Type	Values	Description	Default
	enable	boolean	true / false	Determines whether silent video detection (no faces or movement) is disabled (false) or enabled (true).	true
	silent_after	number	1-120	The minimum number of seconds the participant's video has to silent before it is considered for removal from the video mix (it may take longer than this before the video is actually removed).	15
	require_no_faces	boolean	true/false	Determines whether or not detected faces in the video are taken into consideration: <ul style="list-style-type: none">• true: a participant cannot be marked as silent if a face is detected.• false: face-detection is ignored.	true
	reactivate_after	number	1-5	The minimum number of seconds of moving video that is required before marking the participant as no longer silent.	2

Request example:

```
{"config": {"enable": true, "silent_after": 15, "require_no_faces": true, "reactivate_after": 2}}
```

Response: the result is true if successful, false otherwise.

set_classification_level

This POST request sets the classification level to use from the theme assigned to the conference.

Request example:

```
{"level": 4}
```

Request fields:

level	number	The classification level to use. It must be a valid level key from the theme associated with the conference.
-------	--------	--

Response: the result is true if successful, false otherwise.

get_classification_level

This GET request obtains the set of available classification levels and the currently active level. For example:

```
{
  "status": "success",
  "result": {
    "levels": {
      "0": "Unclassified",
      "1": "Official",
      "2": "Official Sensitive",
      "3": "RESTRICTED",
      "4": "Confidential",
      "5": "Secret",
      "6": "Top Secret"
    },
    "current": 1
  }
}
```

theme

Provides the theme resources of the conference (direct media only). Used in conjunction with the `splash_screen` server event, the relevant theme resources can be used to locally render a particular splash screen on the client.

The theme metadata for a given conference is requested with a GET request to `/theme`

Example response:

```
{
  "status": "success",
  "result": {
    "direct_media_welcome": {
      "layout_type": "direct_media",
      "background": {
        "path": "background.jpg"
      },
      "elements": [
        {
          "type": "text",
          "color": 4294967295,
          "text": "Welcome"
        }
      ]
    },
    ...
  }
}
```

To fetch a particular resource such as an image. A GET request is made to `/theme/<resource path>`

Example request:

GET /api/client/v2/conferences/<conference_alias>/theme/background.jpg

available_layouts

This returns a list of all available layouts for the given conference. This includes the inbuilt layouts plus any custom layouts available on this conference.

Example request:

GET /api/client/v2/conferences/<conference_alias>/available_layouts

An example response to the default theme is:

```
{
  "status": "success",
  "result": [
    "5:7",
    "1:7",
    "2:21",
    "1:21",
    "4:0",
    "9:0",
    "16:0",
    "25:0",
    "1:0",
    "1:33",
    "teams",
    "one_main_twelve_around",
    "two_mains_eight_around"
  ]
}
```

layout_svgs

This provides a list of SVG string representations of the layouts that are active on the given conference.

Example request:

GET: https://{worker}/api/client/v2/conferences/{conference_alias}/layout_svgs

An example response (only the 1:7 layout is shown here) could contain:

```
{
  "status": "success",
  "result": {
    "1:7": "<svg width='140' height='88' viewBox='0 0 140 88' fill='none' xmlns='http://www.w3.org/2000/svg'>\n<rect
x='0.5' y='0.5' width='139' height='87' rx='3.5' fill='currentColor' stroke='#BBBFC3'></rect>\n<rect x='33'
y='12.1044' width='74' height='45.3913' rx='2' fill='#BBBFC3'></rect>\n<rect x='4' y='66' width='18' height='14'
rx='2' fill='#BBBFC3'></rect>\n<rect x='23' y='66' width='18' height='14' rx='2' fill='#BBBFC3'></rect>\n<rect
x='42' y='66' width='18' height='14' rx='2' fill='#BBBFC3'></rect>\n<rect x='61' y='66' width='18' height='14'
rx='2' fill='#BBBFC3'></rect>\n<rect x='80' y='66' width='18' height='14' rx='2' fill='#BBBFC3'></rect>\n<rect
x='99' y='66' width='18' height='14' rx='2' fill='#BBBFC3'></rect>\n<rect x='118' y='66' width='18' height='14'
rx='2' fill='#BBBFC3'></rect></svg>"
  }
}
```

breakouts

Creates a breakout room, and optionally transfers participants into that room.

Example request:

GET /api/client/v2/conferences/<conference_alias>/breakouts

Request fields:

name	string	Name of the breakout room.
duration	number	Optional timer in seconds for how long the breakout is open (not present or 0 = indefinitely).
end_action	string	What to do with participants when the timer expires, or the breakout is closed: <ul style="list-style-type: none">• "disconnect": disconnect them all• "transfer": transfer them back to the main room Default: "transfer"
participants	object	Optional dictionary of current room ids and participants to transfer to the breakout room on creation. Use "main" to represent the main room. For example: <pre>{"main": ["uuid1", "uuid2", "uuid3"], "breakout_uuid1": ["uuid4"]}</pre> transfers participants "uuid1", "uuid2", and "uuid3" from the main room, plus participant "uuid4" from the "breakout_uuid1" breakout room, to the new breakout room.
guests_allowed_to_leave	boolean	Determines whether the guests can return to the main room (before any timer expires). <ul style="list-style-type: none">• true: allows the guest to return to the main room e.g. via a "return" button• false: the guest cannot return Note that this does not prevent the guest leaving by disconnecting and re-joining the same conference.

Response example:

```
{
  "breakout_uuid": "00000000-0000-0000-0000-000000000001"
}
```

The response contains the uuid of the new breakout room.

Participant functions

Within a conference, operations can be performed on participants, if the client has Host privileges.

These participant REST URIs take the format:

`https://<node_address>/api/client/v2/conferences/<conference_alias>/participants/<participant_uuid>/<request>`

where `<node_address>` is the Conferencing Node, `<conference_alias>` is an alias of the conference, and `<participant_uuid>` is the uuid of the participant you are controlling. Under this path comes the request, for example:

`https://10.0.0.1/api/client/v2/conferences/meet_alice/participants/7f8bdd7f-2d39-4c3f-9236-3e95b21f21a8/disconnect`

disconnect

This POST request disconnects a participant.

Request: empty

Response: the result is true if successful, false otherwise.

mute / unmute

These POST requests are used to mute or unmute a participant's audio.

Request: empty.

Response: the result is true if successful, false otherwise.

video_muted / video_unmuted

These POST requests are used to mute or unmute a participant's video.

Request: empty.

Response: the result is true if successful, false otherwise.

allowrxpresentation / denyrxpresentation

These POST requests are used to enable or disable a participant from receiving the presentation stream. (Participants are enabled by default.)

Request: empty.

Response: the result is true if successful, false otherwise.

spotlighton / spotlightoff

These POST requests are used to enable or disable the "spotlight" on a participant.

The spotlight feature locks any spotlighted participants in the primary positions in the stage layout, ahead of any current speakers. When any participants have been spotlighted, the first one to be spotlighted has the main speaker position, the second one has the second position (leftmost small video, for example), and so on. All remaining participants are arranged by most recent voice activity, as usual.

Request: empty.

Response: the result is true if successful, false otherwise.

unlock

This POST request lets a specified participant into the conference from the waiting room of a locked conference.

Request: empty.

Response: the result is true if successful, false otherwise.

dtmf

This POST request sends DTMF digits to the participant.

Request example:

```
{"digits": "1234"}
```

Request fields:

digits	string	The DTMF digits to send.
--------	--------	--------------------------

Response: the result is true if successful, false otherwise.

message

This POST request sends a message to the participant.

Request example:

```
{"type": "text/plain", "payload": "Hello World"}
```

Request fields:

type	string	The MIME Content-Type. This must be "text/plain" or "application/json".
payload	string	The contents of the message.

Response: the result is true if successful, false otherwise.

calls

This POST request upgrades this connection to have an audio/video/presentation call element.

Pexip expects the call to contain three media sections for three streams: audio, video, and presentation. The presentation section should contain an additional attribute "a=content:slides", to indicate this is the presentation stream.

Request example:

```
{"call_type": "WEBRTC", "sdp": "..."} 
```

Request fields:

call_type	string	"WEBRTC" for a WebRTC call.
sdp	string	Contains the SDP of the sender. In direct media call scenarios, the first joining participant can expect an empty SDP in the response indicating that the far-end is not ready to negotiate media (/ack does not need to be called yet). At a later time, the client shall receive a new_offer server event containing the SDP.

fecc_supported	boolean	Set to true if this participant can be sent FECC messages; false if not. Default: false
media_type	string	Indicates the type of media that the client intends to send: <ul style="list-style-type: none">• "video": main video plus presentation• "video-only": main video only• "audio": audio-only Default: "video"
present	string	This optional parameter can be set with value "main" to request presentation, when active, to be received in place of main video rather than as a separate stream. In this case, the SDP only needs to include audio and video streams.

Response example:

```
{ "status": "success", "result": {  
  "call_uuid": "50ed679d-c622-4c0e-b251-e217f2aa030b",  
  "sdp": "..."} }
```

The response contains the SDP of the Pexip node, and a call_uuid. This call_uuid is used to control the call. The [ack](#) function must be called on this call_uuid in order to start media after the SDP has been exchanged and ICE has been completed.

The response may also contain a "turn" array of additional TURN servers to use, if the **Enable media relay on TCP port 443** feature is enabled. A client wanting to use this would need to trigger an ICE restart and send an [update](#) message to update the SDP.

overlaytext

Changes the participant name overlay text. The text is only applied if overlay text is enabled on a VMR. It can also change the text of an audio-only participant.

Request example:

```
{ "text": "The Dude" }
```

Request fields:

text	string	Text to use as the participant name overlay text.
------	--------	---

Response: the result is true if successful, false otherwise.

pres_in_mix

Controls whether or not the participant sees presentation in the layout mix (Adaptive Composition layout only).

Request example:

```
{ "state": true }
```

Request fields:

state	boolean	Controls whether or not the participant sees presentation in the layout mix.
-------	---------	--

Response: the result is true if successful, false otherwise.

role

Changes the role of the participant.

Request example:

```
{"role": "chair"}
```

Request fields:

role	string	"chair" = Host participant; "guest" = Guest participant
------	--------	---

Response: the result is true if successful, false otherwise.

fecc

Send Far End Camera Control messaging to the participant.

Note that this does not send FECC to all participants; it can either be used in a gateway call or be sent to a specific participant identified by the target UUID (as seen in the participant list).

Request fields:

action	string	Either "start", "stop", or "continue".
movement	array	An array of movements, consisting of:
		axis string Either "pan", "tilt", or "zoom".
		direction string Use "left", "right" for pan; "up", "down" for tilt; or "in", "out" for zoom.
		Which means that you could, for example, send a command to pan, tilt and zoom at the same time.
timeout	number	The duration for which to send the signal. Recommended values are 1000 (1 second) for initial "start" message; 200 for "continue" messages.

Request example:

```
{"action": "start", "movement": [{"axis": "pan", "direction": "left"}, {"axis": "zoom", "direction": "in"}], "timeout": 1000};
```

Response: the result is true if successful, false otherwise.

buzz

This POST request raises a participant's hand.

Request: empty

Response: the result is true if successful, false otherwise.

clearbuzz

This POST request lowers a participant's hand.

Request: empty

Response: the result is true if successful, false otherwise.

transfer

Transfers a participant to another conference.

The target conference is identified by the alias in "conference_alias", and they will have the specified "role". If the target is PIN-protected, the PIN for the target role must be specified in the "pin" field.

Request example:

```
{"role": "guest", "conference_alias": "meet@example.com", "pin": "1234"}
```

Request fields:

role	string	Role can be "guest" or "chair" (Host).
conference_alias	string	Target conference alias.
pin	string	PIN code for the specified role at the specified conference, if required.

Response: the result is true if successful, false otherwise.

take_floor

This POST request signals that the participant is starting sending full-motion presentation.

Request: empty.

Response: a return code other than 200 indicates that the request for presentation has been denied.

release_floor

This POST request signals that the participant has finished sending full-motion presentation.

Request: empty.

Response: the result is true if successful, false otherwise.

avatar.jpg

This GET request obtains the image to display to represent a conference participant or directory contact.

statistics

Report participant media statistics to Pexip Infinity (direct media only). The client should notify at the period specified by the `client_stats_update_interval` in the `request_token` response.

Example request:

```
{
  "audio": {
    "rx_bitrate": 64.4,
    "rx_codec": "opus",
    "rx_jitter": 2,
    "rx_packets_lost": 0,
    "rx_packets_received": 914,
    "tx_bitrate": 63.728,
    "tx_codec": "opus",
    "tx_packets_sent": 914,
    "tx_rb_jitter": 2,
    "tx_rb_packetslost": 0
  },
  "video": {
```



```
    "rx_bitrate": 2215.744,
    "rx_codec": "VP8",
    "rx_fps": 21,
    "rx_packets_lost": 0,
    "rx_packets_received": 4206,
    "rx_resolution": "1280x720",
    "tx_bitrate": 2089.296,
    "tx_codec": "VP8",
    "tx_packets_sent": 3633,
    "tx_rb_packetslost": 0,
    "tx_resolution": "1280x720"
  },
  "presentation": {
    "rx_packets_lost": 0,
    "rx_packets_received": 0,
    "tx_bitrate": 989.304,
    "tx_codec": "VP8",
    "tx_packets_sent": 1714,
    "tx_rb_packetslost": 0,
    "tx_resolution": "1792x1120"
  }
}
```

preferred_aspect_ratio

Specifies the aspect ratio the participant would like to receive.

Request fields:

aspect_ratio	float	A number between 0 and 2 representing the aspect ratio the participant would like to receive. For example 9 / 16 would be 0.5625.
--------------	-------	---

Response: the result is true if successful, false otherwise.

Breakout rooms functions

These breakout room REST URIs typically take the format:

`https://<node_address>/api/client/v2/conferences/<conference_alias>/breakouts/<breakout_uuid>/<request>`

Breakout room controls

Each breakout room can be controlled by hosts like a normal VMR e.g. to mute all guests, transform_layout etc.

For example:

```
POST /api/client/v2/conferences/meet.alice/breakouts/00000000-0000-0000-0000-000000000001/transform_layout
POST /api/client/v2/conferences/meet.alice/breakouts/00000000-0000-0000-0000-000000000001/muteguests
POST /api/client/v2/conferences/meet.alice/breakouts/00000000-0000-0000-0000-000000000001/set_classification_level
```

Closing a breakout room

To disconnect/close a breakout room:

```
POST /api/client/v2/conferences/<conference_alias>/breakouts/<breakout_uuid>/disconnect
```

Depending on the end_action specified on creation, participants will be transferred back to the main room or disconnected.

Emptying all breakout rooms

To empty all breakout rooms:

```
POST /api/client/v2/conferences/<conference_alias>/breakouts/empty
```

This moves all participants in all rooms back to the main room, and leaves the rooms open (i.e. it leaves the Host API control participants present).

Ask for help / dismiss help request

The `breakoutbuzz` POST request requests help for the breakout room:

```
POST /api/client/v2/conferences/<conference_alias>/breakoutbuzz
```

The resulting `conference_update` event contains a `breakoutbuzz` field with a `value` (true/false) and a `time` which shows when it was set.

Anyone in the breakout room, or a Host, can use the `clearbreakoutbuzz` POST request to cancel the request for help:

```
POST /api/client/v2/conferences/<conference_alias>/clearbreakoutbuzz
```

or

```
POST /api/client/v2/conferences/<conference_alias>/breakouts/<breakout_uuid>/clearbreakoutbuzz
```

Transfer participants back to the main room

To transfer participants from a breakout room back to the main room:

```
POST /api/client/v2/conferences/<conference_alias>/breakouts/<breakout_uuid>/participants/breakout
```

For example:

```
POST /api/client/v2/conferences/meet.alice/breakouts/00000000-0000-0000-0000-000000000001/participants/breakout
{
  "breakout_uuid": "main",
  "participants": ["uuid1", "uuid2", "uuid3"]
}
```

where `uuid1`, `uuid2`, `uuid3` are valid participant uids in breakout room `00000000-0000-0000-0000-000000000001`.

- If the `participants` list is empty then all participants are moved.
- You can transfer between breakout rooms by specifying the breakout room uuid instead of "main" in the above request.
- You can transfer participants back to their previous room by specifying a breakout room uuid of "previous". For example:

```
{
  "breakout_uuid": "previous",
```

```
"participants": []  
}
```

would move all participants back to their previous rooms.

(If a participant does not have a previous room, then the transfer will just fail.)

Move participant back to the main room

If guests are allowed to leave the breakout room, a participant can use `leavebreakout` to transfer back to the main room:

```
POST /api/client/v2/conferences/<conference_alias>/leavebreakout
```

Other participant actions

You can also perform actions on specific participants in the breakout room, for example:

```
POST /api/client/v2/conferences/meet.alice/breakouts/<breakout_uuid>/participants/<participant_uuid>/mute
```

```
POST /api/client/v2/conferences/meet.alice/breakouts/<breakout_uuid>/participants/<participant_uuid>/overlaytext
```

Note that disconnecting a participant from a breakout room will disconnect them entirely and not move them back to the main room.

Call functions

Using the `call_uuid`, further operations can be undertaken on the calls as part of the nominated participant.

These call REST URIs take the format:

`https://<node_address>/api/client/v2/conferences/<conference_alias>/participants/<participant_uuid>/calls/<call_uuid>/<request>`

where `<node_address>` is the Conferencing Node, `<conference_alias>` is an alias of the conference, `<participant_uuid>` is the uuid of the participant, and `<call_uuid>` is the uuid of the call you are controlling. Under this path comes the request, for example:

`https://10.0.0.1/api/client/v2/conferences/meet_alice/participants/7f8bdd7f-2d39-4c3f-9236-3e95b21f21a8/calls/c34f35f-1060-438c-9e87-6c2dffc9980/disconnect`

ack

This POST request starts media for the specified call (WebRTC calls only).

Request: empty.

For direct media calls only: on receiving a `new_offer` server event, containing the remote SDP, `ack` should be called with the local SDP, to complete the offer/answer exchange. Example request:

```
{"sdp": "..."} 
```

Response: the result is true if successful, false otherwise.

disconnect

This POST request disconnects the specified call.

Request: empty.

Response: the result is true if successful, false otherwise.

dtmf

For a gateway call only, this POST request sends DTMF digits to the remote participant.

Request example:

```
{"digits": "1234"} 
```

Response: the result is true if successful, false otherwise.

new_candidate

This POST request sends a new ICE candidate if doing trickle ICE.

Request example:

```
{"candidate": "candidate:1732786348 1 udp 2124262783 2a02:c7f:615::eration 0 ufrag YAeD network-id 2 network-cost 10", "mid": "0", "ufrag": "YAeD", "pwd": "IfZniTlYHipJXEg4quoI00ek"}
```

Request fields:

candidate	string	Representation of address in <code>candidate-attribute</code> format as per RFC5245.
mid	string	The media stream identifier tag.
ufrag	string	The randomly generated username fragment of the ICE credentials.
pwd	string	The randomly generated password of the ICE credentials.

Response: the result is true if successful, false otherwise.

update

This POST request sends a new SDP.

Request example:

```
{"sdp": "..."} 
```

Request fields:

sdp	string	The new SDP.
fecc_supported	boolean	Set to true if this participant can be sent FECC messages; false if not. Default: false

Response example:

```
{"status": "success", "result": "..."} 
```

fecc

For a gateway call only, this POST request sends FECC messages to the remote participant.

Request example:

```
{"action": "start", "movement": [{"axis": "pan", "direction": "left"}, {"axis": "zoom", "direction": "in"}],
"timeout": 1000};
```

Request fields:

action	string	Either "start", "stop", or "continue".		
movement	array	An array of movements, consisting of:		
		axis	string	Either "pan", "tilt", or "zoom".
		direction	string	Use "left", "right" for pan; "up", "down" for tilt; or "in", "out" for zoom.
		Which means that you could, for example, send a command to pan, tilt and zoom at the same time.		
timeout	number	The duration for which to send the signal. Recommended values are 1000 (1 second) for initial "start" message; 200 for "continue" messages.		

Response: the result is true if successful, false otherwise.

Server-sent events

Clients can subscribe to an HTTP EventSource which feeds events from the conference as they occur.

To subscribe, open an HTTP connection to:

`https://<node_address>/api/client/v2/conferences/<conference_alias>/events?token=<token_id>`

where **<node_address>** is the Conferencing Node, **<conference_alias>** is an alias of the conference, and **<token_id>** is the session token, for example:

`https://10.0.0.1/api/client/v2/conferences/meet_alice/events?token=123456`

This allows the token to be specified on the URI, since custom headers cannot be added to Event Sources in browsers today. However, if headers can be added this will be accepted too, and the query parameter will not be required.

Each event contains an event name, and some events may contain a payload of data, which is a JSON object.

presentation_start

This marks the start of a presentation, and includes the information on which participant is presenting, and the presentation source ("video" or "static").

Example data:

```
{"presenter_name": "Bob", "presenter_uri": "bob@example.com", "source": "video"}
```

presentation_stop

The presentation has finished.

Data: none

presentation_frame

A new presentation frame is available at:

`https://<node_address>/api/client/v2/conferences/<conference_alias>/presentation.jpeg`

An alternative image at a higher resolution is also available at:

`https://<node_address>/api/client/v2/conferences/<conference_alias>/presentation_high.jpeg`

Note that these URLs require the token and the event ID of the **presentation_frame** event to be present as a header or a query parameter in order to download the presentation frame, for example:

`https://10.0.0.1/api/client/v2/conferences/meet_alice/presentation.jpeg?id=MTAuNDQuOTkuMl8xOA==&token=b3duZXI9T...etc...2FmGzA%3D`

Data: none

participant_create

A new participant has joined the conference.

The JSON object fields include:

buzz_time	number	A Unix timestamp of when this participant raised their hand, otherwise zero.
call_direction	string	Either "in" or "out" as to whether this is an inbound or outbound call.
call_tag	string	An optional call tag that is assigned to this participant.
disconnect_supported	string	Set to "YES" if the participant can be disconnected, "NO" if not.
display_name	string	The display name of the participant.
encryption	string	"On" or "Off" as to whether this participant is connected via encrypted media.
external_node_uuid	string	The UUID of an external node e.g. a Skype for Business / Lync meeting associated with an external participant. This allows grouping of external participants as the UUID will be the same for all participants associated with that external node.
fecc_supported	string	Set to "YES" if this participant can be sent FECC messages; "NO" if not.
has_media	boolean	Boolean indicating whether the user has media capabilities.
show_live_captions	boolean	Boolean indication whether the current participant is receiving live captions.
is_audio_only_call	string	Set to "YES" if the call is audio only.
is_external	boolean	Boolean indicating if it is an external participant, e.g. coming in from a Skype for Business / Lync meeting.
is_idp_authenticated	boolean	Indicates whether the participant has been authenticated with an Identity Provider.
is_main_video_dropped_out	boolean	Boolean indicating if video from the user has been lost.
is_muted	string	Set to "YES" if the participant is administratively audio muted.
is_presenting	string	Set to "YES" if the participant is the current presenter.
is_streaming_conference	boolean	Boolean indicating whether this is a streaming/recording participant.
is_video_call	string	Set to "YES" if the call has video capability.
is_video_muted	boolean	Boolean indicating whether this participant has muted their video.
is_video_silent	boolean	Boolean indicating if the user has moved away (silent video detection in an Adaptive Composition layout).
local_alias	string	The calling or "from" alias. This is the alias that the recipient would use to return the call.
mute_supported	string	Set to "YES" if the participant can be muted, "NO" if not.
overlay_text	string	Text that may be used as an alternative to display_name as the participant name overlay text.
presentation_supported	string	Set to "YES" if the participant can send presentation, "NO" if not.

protocol *	string	The call protocol. Values: "api", "webrtc", "sip", "rtmp", "h323" or "mssip". (Note that the protocol is always reported as "api" when a Connect app dials in to Pexip Infinity.)
role	string	The level of privileges the participant has in the conference: <ul style="list-style-type: none">• "chair": the participant has Host privileges• "guest": the participant has Guest privileges
rx_presentation_policy	string	Set to "ALLOW" if the participant is administratively allowed to receive presentation, or "DENY" if disallowed.
service_type	string	The service type: <ul style="list-style-type: none">• "connecting": for a dial-out participant that has not been answered• "waiting_room": if waiting to be allowed to join a locked conference• "ivr": if on the PIN entry screen• "conference": if in a VMR• "lecture" if in a Virtual Auditorium• "gateway": if it is a gateway call• "test_call": if it is a Test Call Service
spotlight	number	A Unix timestamp of when this participant was spotlighted, if spotlight is used.
start_time	number	A Unix timestamp of when this participant joined (UTC).
transfer_supported	string	Set to "YES" if this participant can be transferred into another VMR; "NO" if not.
uuid	string	The UUID of this participant, to use with other operations.
uri *	string	The URI of the participant.
vendor *	string	The vendor identifier of the browser/endpoint with which the participant is connecting.
* Empty for Guest participants.		

Example data:

```
{
  "buzz_time": 0,
  "call_direction": "in",
  "call_tag": "def456",
  "disconnect_supported": "YES",
  "display_name": "Alice",
  "encryption": "On",
  "external_node_uuid": "",
  "fecc_supported": "NO",
  "has_media": false,
  "is_audio_only_call": "NO",
  "is_external": false,
  "is_idp_authenticated": false,
  "is_muted": "NO",
  "is_presenting": "NO",
  "is_streaming_conference": false,
  "is_video_call": "YES",
  "is_video_muted": false,
  "local_alias": "meet.alice",
  "mute_supported": "YES",
  "overlay_text": "Alice",
  "presentation_supported": "NO",
  "protocol": "api",
  "role": "chair",
  "rx_presentation_policy": "ALLOW",
  "service_type": "conference",
```

```
"spotlight": 0,
"start_time": 1441720992,
"transfer_supported": "YES",
"uri": "Infinity_Connect_10.44.21.35",
"uuid": "50b956c8-9a63-4711-8630-3810f8666b04",
"vendor": "Pexip Infinity Connect/2.0.0-25227.0.0 (Windows NT 6.1; WOW64) nwjs/0.12.2 Chrome/41.0.2272.76"
}
```

participant_update

A participant's properties have changed.

Data: a full JSON object is supplied, as for [participant_create](#).

participant_delete

A participant has left the conference.

Data: the JSON object contains the UUID of the deleted participant, for example:

```
{"uuid": "65b4af2f-657a-4081-98a8-b17667628ce3"}
```

participant_sync_begin / participant_sync_end

At the start of the EventSource connection, these two messages start and end the sending of the complete participant list in the form of [participant_create](#) events. This allows a participant that has been temporarily disconnected to re-sync the participant list.

conference_update

Conference properties have been updated. For example:

```
{"locked": false, "guests_muted": false, "all_muted": false, "presentation_allowed": true, "started": true,
"livecaptions_available": true, "direct_media": false, "breakout": true, "breakout_name": "My Breakout Room",
"breakout_description": "Quick discussion about XYZ", "end_action": "transfer", "end_time": 1688688005, "guests_
allowed_to_leave": false}
```

layout

The stage layout has changed.

Data: an object containing the following fields:

view	string	<p>The layout currently seen by the participant:</p> <ul style="list-style-type: none">"1:0": main speaker only"1:7": main speaker and up to 7 previous speakers"1:21": main speaker and up to 21 previous speakers"2:21": 2 main speakers and up to 21 previous speakers"1:33": 1 small main speaker and up to 33 other speakers"4:0": 2x2 layout, up to a maximum of 4 speakers"9:0": 3x3 layout, up to a maximum of 9 speakers"16:0": 4x4 layout, up to a maximum of 16 speakers"25:0": 5x5 layout, up to a maximum of 25 speakers"one_main_nine_around": one main speaker and up to 9 other participants"one_main_twelve_around": large main speaker and up to 12 other participants"two_mains_eight_around": two main speakers and up to 8 other participants"5:7": Adaptive Composition (AC) layout"ac_presentation_in_mix": AC and viewing a presentation in the layout mix (single person presenter)"ac_presentation_in_mix_group": AC and viewing a presentation in the layout mix (group presenter)"teams": Teams-like layout (this is only suitable for use with Microsoft Teams gateway calls)
participants	array	An array of UUIDs for the participants, in order, starting from the main speaker position. Note that an API-only participant (no audio or video) always receives an empty participants UUID list.
requested_layout	string	Indicates the layout requested by the last <code>transform_layout</code> API request. For example, when using a Virtual Auditorium as a Host, the view is "2:21" whereas <code>requested_layout</code> can be <code>{"primary_screen": {"chair": "2:21", "guest": "4:0"}}</code> .
overlay_text_enabled	boolean	Indicates whether overlay text is in use.

Example data:

```
{ "view": "1:7",  
  "participants": ["a0196175-b462-48a1-b95c-f322c3af57c1", "65b4af2f-657a-4081-98a8-b17667628ce3"],  
  "requested_layout": {"primary_screen": {"chair": "2:21", "guest": "4:0"}},  
  "overlay_text_enabled": false }
```

message_received

A chat message has been received.

Data: an object containing the following fields:

origin	string	Name of the sending participant.
uuid	string	UUID of the sending participant.
type	string	MIME content-type of the message. Either "text/plain" or "application/json".
direct	boolean	Indicates if this message was sent direct to the participant (true) or broadcast to the conference (false).
payload	string	Message contents.

Example data:

```
{ "origin": "Alice",  
  "uuid": "eca55900-274d-498c-beba-2169aad9ce1f",  
  "type": "text/plain",  
  "direct": false,  
  "payload": "Hello World" }
```

stage

An update to the "stage layout" is available. This declares the order of active speakers, and their voice activity.

Data: an array of objects per active participant. Each participant has the following fields:

participant_uuid	string	The UUID of the participant.
stage_index	number	The index of the participant on the "stage". 0 is most recent speaker, 1 is the next most recent etc.
vad	number	Audio speaking indication. 0 = not speaking, 100 = speaking.

Example data:

```
[  
  { "stage_index": 0,  
    "participant_uuid": "a0196175-b462-48a1-b95c-f322c3af57c1",  
    "vad": 0 },  
  { "stage_index": 1,  
    "participant_uuid": "65b4af2f-657a-4081-98a8-b17667628ce3",  
    "vad": 0 }  
]
```

call_disconnected

This is sent when a child call has been disconnected (e.g. when a screensharing child call has been closed if presentation has been stolen by another participant).

Data: contains both the UUID of the child call being disconnected, and the reason for the disconnection if available, e.g.:

```
{ "call_uuid": "50ed679d-c622-4c0e-b251-e217f2aa030b",  
  "reason": "API initiated participant disconnect" }
```

disconnect

This is sent when the participant is being disconnected from the Pexip side.

Data: the reason parameter contains a reason for this disconnection, if available, e.g.:

```
{"reason": "API initiated participant disconnect"}
```

splash_screen

The client should display a splash screen. For VMRs with direct media enabled, it is the clients' responsibility to render local screens. The `screen_key` uniquely identifies the type of screen the client should display. On receiving the event with no data, the splash screen should be cleared.

Optionally the client can implement these local splash screens by fetching the theme resources from Pexip Infinity. See [theme](#) for more information.

Example data:

```
{"screen_key": "direct_media_welcome"} | None
```

The screen keys may be `direct_media_welcome`, `direct_media_waiting_for_host`, `direct_media_other_participants_audio_only`, `direct_media_escalate`, or `direct_media_deescalate`.

new_offer

Received a SDP offer from the far-end participant (direct media only). The far-end has sent an updated SDP via `/calls` for the client to consider. The `/ack` function must be called in order to start media after the SDP has been exchanged and ICE has been completed.

Example data:

```
{"sdp": "..."} 
```

update_sdp

Received a SDP update from the far-end participant (direct media only). The far-end has sent an updated SDP via `/update` for the client to consider. Typically received for purposes of an ICE restart.

Example data:

```
{"sdp": "..."} 
```

new_candidate

Received a new ICE candidate from the far-end participant (direct media only). The far-end has sent a new ICE candidate if doing trickle ICE via `/new_candidate` for the client to consider.

Example data:

```
{"candidate": "candidate:1732786348 1 udp 2124262783 2a02:c7f:615::eration 0 ufrag YAEd network-id 2 network-cost 10", "mid": "0", "ufrag": "YAEd", "pwd": "IfZniTlYHipJXEg4quoI00ek"}
```

peer_disconnect

Indicates that the far-end participant has disconnected (direct media only).

Example data: None

refer

The client should connect to a different conference (alias).

Example data:

```
{"alias": "meet.bob", "breakout_name": "Room 1"}
```

breakout_begin

Hosts (including the creator) receive a **breakout_begin** event which contains information about a new breakout room, plus the participant uuid that they have been allocated inside the breakout as a control participant.

Example data:

```
{"breakout_uuid": "00000000-0000-0000-0000-000000000001", "participant_uuid": "participantxyz"}
```

Hosts (including the creator) will now receive [breakout events](#) on the main room EventSource.

breakout_event

Hosts (including the creator) receive breakout events on the main room EventSource. These are normal events that are occurring within the breakout VMR — so you can build a roster list, see the state of the conference etc.

Example data:

```
{"breakout_uuid": "00000000-0000-0000-0000-000000000001", "event": "participant_sync_begin", "data": null}
{"breakout_uuid": "00000000-0000-0000-0000-000000000001", "event": "participant_create", "data": {"uuid": "1234",
"uri": "user@domain.com", ...}}
```

breakout_end

Hosts receive a **breakout_end** event to tell them when a breakout room is no longer available.

Example data:

```
{"breakout_uuid": "00000000-0000-0000-0000-000000000001", "participant_uuid": "participantxyz" }
```

where the **participant_uuid** represents that host's participant in the breakout room.

breakout_refer

Sent when a participant specified in the participant lists is currently a host (and hence has an API participant in the breakout). This tells them to move their media to the breakout room specified by the uuid in the event.

Example data:

```
{"breakout_uuid": "main"}
{"breakout_uuid": "00000000-0000-0000-0000-000000000001", "breakout_name": "Room 2"}
```

Other miscellaneous requests

Conferencing Node status (maintenance mode)

Load balancers can use the `https://<node_address>/api/client/v2/status` REST API command to check whether a Conferencing Node is in maintenance mode, for example:

`https://10.0.0.1/api/client/v2/status`

If the node **is not** in maintenance mode, it returns a **200 OK** with the following JSON:

```
{
  "status": "success",
  "result": "OK"
}
```

If the node **is** in maintenance mode, it returns a **503** with the following JSON:

```
{
  "status": "failed",
  "result": "Maintenance mode"
}
```

Changelog

Changes in version 34:

- New `layout_svgs` conference control function.
- New `breakouts` functions: `breakoutbuzz`, `clearbreakoutbuzz` and `leavebreakout`.

Changes in version 33:

- New `available_layouts` conference control function.
- New `preferred_aspect_ratio` participant function.
- New functions and events to support breakout rooms:
 - New `breakouts` conference control function.
 - New server events: `breakout_begin`, `breakout_event`, `breakout_end` and `breakout_refer`.
 - New breakout room related parameters in `conference_status` function and `conference_update` event.

Changes in version 32:

- New `overlay_text_enabled` field in the `layout` event.
- New `is_idp_authenticated` parameter in the participant list / roster object.

Changes in version 31:

- New `set_classification_level` and `get_classification_level` conference functions.
- New `theme` conference control function.
- New `message` participant function.
- New `statistics` participant function.
- New server sent events: `splash_screen`, `new_offer`, `update_sdp`, `new_candidate`, `peer_disconnect`.
- The `request_token` request has a new `direct_media` request field and new responses: `direct_media`, `use_relay_candidates_only`, `pex_datachannel_id`, `client_stats_update_interval`.
- New `direct_media` parameter in `conference_status` conference function.
- New `direct_media` parameter in `conference_update` event.
- New `media_type` field in `calls` participant function.
- New `ac_presentation_in_mix` field on `transform_layout` function.
- New `direct` parameter in `message_received` event.

Changes in version 30:

- New `source` field in the `presentation_start` event.
- New `plus_n_pip_enabled` field on `transform_layout` function.
- New `requested_layout` field in the `layout` event.

More information

For more information about using this API, contact your Pexip authorized support representative.