

Blink Qt — WebRTC video conferencing

How Blink Qt joins SylkServer conference rooms with audio and video over WebRTC, next to the MSRP chat it always had, and how it joins the rooms of other SylkServer domains.

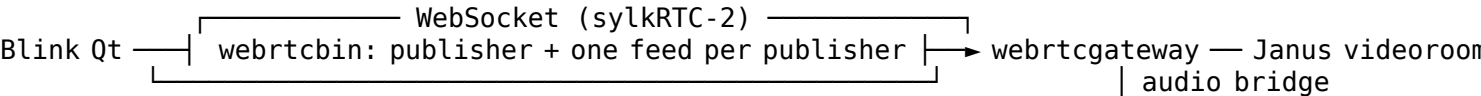
Code	blink/webrtc/ (__init__.py, sylkrtc.py, videoroom.py, videoroomwindow.py, sharedialog.py, speakerdialog.py, infopanel.py, extravideo.py, camerafinder.py, camerasetup.py); hooks in sessions.py, chatwindow.py, mainwindow.py, screen_share.py, videowindow.py, widgets/video.py, __init__.py, logging.py
Server side	SylkServer webrtcgateway (sylkRTC protocol, Janus videoroom), conference application (MSRP switch)
Related	sylkserver ([MSRP] port), opensips-config-sylkserver (CONFERENCE_SERVER_PUBLIC, Contact rewrite), sylk-suite (MSRP port, TLS certificate, source mode)
State	2026-10-24, first working version: own domain and a foreign domain (domain.sylk.link) tested. 2026-10-25: layouts, carousel, speaker selection, moderation, screen sharing, video loss; §10 lists every change since the first version, for the Cocoa port. 2026-10-26: several video feeds per participant (cameras, screen, RTSP cameras), Join panel video sources and profiles, Conference info window; §11 lists them

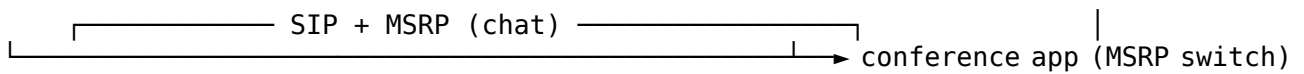
1. Overview

A SylkServer room is two things that share one name:

- the **video room**: a Janus videoroom (an SFU) behind the webrtcgateway, reached over a WebSocket with the sylkRTC protocol. Every participant publishes its camera and microphone on one PeerConnection and subscribes to each other publisher on one PeerConnection per publisher (a *feed*);
- the **MSRP switch**: the conference application, reached over SIP, carrying the room’s chat (MSRP), its conference-info (participants) and, for SIP callers, the mixed audio. The gateway’s audio bridge (sylk-janus-audio-bridge) joins the two.

Blink Qt joins the video room for **audio and video** and the MSRP switch for **chat only**. Sending audio on both would loop the room’s audio through it; the MSRP window therefore never adds audio while a video room is joined (§5). Joining the video room, however it is started (the Join panel, a contact’s Video Call or the header Video button), removes SIP audio and video from a session already connected to the same room, or hangs the session up when that was all it had. A session that keeps its chat is reused, not opened again.





2. Loading and availability

Nothing of blink/webrtc beyond the package's `__init__.py` is imported until a SylkServer is detected: `__init__.py` holds the light checks (`ws_server`, `conference_video_available`, `room_video`, `active_videoroom`, ...) and imports `videoroom` (GStreamer) and `sylkrtc` (Qt WebSockets) only after them.

Check	Where	Meaning
<code>ws_server(account)</code>	<code>__init__.py</code>	the wsServer of the account's domain, from SylkServer discovery (<code>_sylkserver.<domain> TXT</code> → configuration JSON)
<code>media_missing()</code>	<code>videoroom.py</code>	the missing pieces, as (what, Debian package) pairs
<code>media_available()</code>	<code>__init__.py</code>	<code>media_missing()</code> empty; checked once a run
<code>conference_video_available(account).py</code>	<code>__init__.py</code>	both of the above

Required: Qt WebSockets, GStreamer introspection (Gst, GstWebRTC, GstSdp) and the elements `webrtcbin` `nicesrc` `appsrc` `appsink` `decodebin` `videoconvert` `videoscale` `videorate` `audioconvert` `audioresample` `audiomixer` `volume` `opusenc` `rtpopuspay` `autoaudiosrc` `autoaudiosink`, plus one video encoder and payload pair. Optional: `webrtcdsp` + `webrtcchoprobe` (echo cancellation).

Debian packages (Recommends of blink): `gststreamer1.0-plugins-bad`, `gststreamer1.0-nice`, `gststreamer1.0-pulseaudio`, `gir1.2-gst-plugins-bad-1.0`, `python3-pyqt6.qtwebsockets`.

Windows (MSYS2 UCRT64, windows/): `qt6-websockets`, `libnice` and `gst-plugins-bad` are installed by `windows.yml`. `blink.spec` lists `PyQt6.QtWebSockets`, `gi.repository.GstSdp` and `gi.repository.GstWebRTC` as hidden imports so that `PyInstaller` bundles the module and the `typelibs`. When something is missing, the Join panel names it instead of giving the `apt install` hint. Not yet tested on Windows.

The Activity log says what was found, once per change of account, server or missing set:

```
[webrtc] Video conferences available for ag@sylk.link through wss://.../webrtcgateway/ws
[webrtc] WebRTC media: GStreamer 1.24.2, video VP9 (vp9enc), VP8 (vp8enc), H264 (x264enc), audio
[webrtc] Video conferences for ... need: nicesrc (install gststreamer1.0-nice)
```

3. Signalling: `sylkrtc.py`

`SylkRTCConnection` is one `QWebSocket` per `wsServer` URL (`SylkRTCConnections`), in the GUI thread, sub-protocol `sylkRTC-2`.

Constant	Value	
<code>CONNECT_TIMEOUT</code>	15 s	from opening to ready-event
<code>REQUEST_TIMEOUT</code>	20 s	a request without ack/error fails
<code>PING_INTERVAL</code>	30 s	WebSocket ping

- `request(data, callback)` adds a transaction, queues until ready, calls `callback(error)` on ack (`None`) or error (the reason).

- Events (account-event, session-event, videoroom-event) go out by the received signal (not event, which would shadow QObject.event()).
- add_account(account) / add_identity(id, ha1, display_name): account-add with the HA1 of the SIP credentials, as sylkrtrc.js sends it. The account is **never registered** through the gateway: Blink registers over SIP itself, a second registration would fork calls and messages to the gateway.
- TLS certificate errors are ignored when SIP TLS verification is off (tls.verify_server).
- A connection nobody uses is closed one second after its last room ends.
- **Gateway features:** the ready-event carries features (a list of strings) since 2026-10-26: sip-instance (the sip_instance join field is understood) and multiple-video-feeds (child publishers, §4.8). An older gateway sends none and ignores request fields it does not know. Each connection keeps them (connection.features) and they are remembered per wsServer URL (ui.conf webrtc/gateway_features). known_features(url) answers from a live connection or what was seen last; probe_features(url, callback) opens a short connection of its own, reads the ready-event and closes it (the Join panel asks every time it opens).

4. Media: videoroom.py

4.1 Session flow

```

account-add → build publisher, create offer → videoroom-join(offer)
  ← session-state progress / accepted(answer) → link the video encoder, set answer
  ← initial-publishers / publishers-joined → per publisher: videoroom-feed-attach
  ← feed-attached(offer) → feed webrtcbin, answer → videoroom-feed-answer
  ← session-state established, feed-established
leave: videoroom-leave

```

ICE candidates go out as videoroom-session-trickle (end of candidates: an empty list), held back until the request that creates their session on the server (join, feed answer) has been sent.

PUBLISHER_TIMEOUT (20 s from the offer) ends a join that never reaches *established*.

4.2 Pipeline

One GStreamer pipeline per room, so the echo canceller sees what is played:

```

camera (SDK frames, BGRA or I420) → appsrc → convert/scale (borders)/rate → caps (640x480 24 fps)
microphone → webrtcdsp → volume (mute) → level (speaking) → opusenc → rtpopuspay

```

```

webrtcbin (feed) → decodebin (depayloaders without header extensions) → video: convert → app
                                     ↳ audio: convert/resample → level (speaking)

```

- **Camera:** frames of the SDK's camera (SIPApplication.video_device) read through a Frame-BufferVideoRenderer, like any other view of it, so the camera chosen in Blink, the screen included, is what the room sees. "Camera off" sends black frames. The frames come in the producer's own format (BGRA from cameras; the appsrc also takes I420) and are re-read when the device changes (VideoDeviceDidChangeCamera).
- **Our video SSRC is announced in the offer:** the encoder chain is only built after the answer, so webrtcbin would offer no a=ssrc for video, and Janus drops RTP from an SSRC it was not told (the symptom: our video sent, no RTCP report back, nobody sees us). The SSRC is chosen before the offer, put in the codec preferences (ssrc=(uint)N, which webrtcbin offers) and given to the payload. Specific to GStreamer: libwebrtc announces its SSRCs.
- **Screen:** while the video device is the screen, the encoder caps go to 1280×720 at 15 fps and the bitrate to 1800 kbit/s, renegotiated in place (SCREEN_CHANGES_SIZE); back to 640×480 at 24 fps, 1024 kbit/s for a camera. 720p is the largest frame of H264 level 3.1 (42e01f), what the rooms negotiate.
- **Codecs:** the room's video codec is set on the server (Janus videocodec), so the offer lists every codec that can be encoded here (VP9 98, VP8 96, H264 102 with x264enc oropenh264enc)

and the encoder for the one the answer picked is linked afterwards; the transceiver's codec preferences are then narrowed to that codec (the encoder's output would not match the offered profile-level-id). Audio: Opus 111, OPUS/48000/2, given as codec preferences because the offer is built before the microphone has caps.

- **Bitrate:** 1024 kbit/s video, 32 kbit/s Opus with in-band FEC.
- **No GLib main loop:** the bus is read with a sync handler; everything that touches the signalling or the UI is handed to the GUI thread.
- **Orientation (CVO):** a mobile sends the camera as it is held and says how it is turned in the RTP header extension urn:3gpp:video-orientation. Its id is taken from each feed's offer and added to the answer (webrtcbin drops extensions it does not implement); a pad probe reads it from the RTP packets and the decoded picture is turned clockwise by 0/90/180/270°. The same probe counts each feed's video bytes (§4.6).
- **Depayloaders:** decodebin's depayloaders have auto-header-extension turned off: after a feed's video is paused and resumed one could drop a frame holding extension buffers and GStreamer aborts (gststrtpbasedepayload.c: hdrext_buffers == 0).
- **Our MSRP leg is us:** the MSRP INVITE to a room carries our +sip.instance (as on REGISTER) and videoroom-join sends the same as sip_instance; the gateway then knows the focus endpoint with that participant_id is this device and lists it neither as a second participant nor as a SIP caller.

4.3 Participants

initial-publishers / publishers-joined / publishers-left / publisher-updated.

Publisher type	Subscribed	Tile
webrtc	yes	picture and name
bridge (the audio bridge: the SIP callers' mixed audio)	yes (audio)	none
sip (a SIP caller behind the bridge)	no	round avatar in the carousel
sip with our own address (our MSRP chat leg)	no	none (is_self)
webrtc whose feed offer has no usable video (m=video rejected, inactive, recvonly)	yes	round avatar in the carousel

- **Name:** the name of the contact with that address in our contacts list, else the room's display name, else the address.
- **Video:** Participant.video is None until the feed's offer, then whether it carries video; a SIP caller has none. Audio only participants never get a grid tile.
- A moderator's mute-request mutes the microphone.

4.4 Room chat over WebRTC

send_chat_message() (videoroom-message) and the message / message-delivery events, posted as VideoroomSessionGotMessage / VideoroomSessionDidDeliverMessage. The gateway relays them to the room's MSRP switch. Used by the MSRP window when its own MSRP session to the room is not connected (§5.3).

4.5 Room events and requests

As Sylk Mobile's react-native-sylkrtc (sylk-desktop's bundled sylkrtc is older).

Request	Payload	Event back (videoroom-event)
videoroom-configure	active_participants: 0-2 publisher ids, ours = our session id	configure {active_participants, originator (an id, ours, or videoroom)}, to everyone, also on join
videoroom-toggle-hand	[session_id: someone else's, to lower it]	raised-hands {raised_hands: ids, ours = our session id}
videoroom-mute-audio-participants	—	mute-audio {originator} to everyone: the others mute their microphone; the one who asked ignores it
videoroom-mute-participant	participant_id, muted	the WebRTC target gets mute-request (§4.3); a SIP one is muted at the mix (REFER ;method=BYE to the focus)
videoroom-remove	participants: SIP URIs	invite-status
videoroom-invite	participants	{participant, state progress/success/failed, code, reason}
videoroom-session-update (on a feed)	options: {video: false/true}	— (the server stops/resumes that feed's video)
—	—	conference-audio-levels {levels: [{participant_id, rx, rx_peak, ...}]}: SIP participants' levels, the mixer's 0-255 scale
videoroom-join	+ sip_instance: our SIP stack's instance-id (optional)	—
videoroom-join (an extra feed)	+ parent: our publisher's session id, label: its name; audio false	the feed is a publisher shown under us (§4.8)

4.6 Speaking, video loss, pausing

- **Speaking:** a GStreamer level element in each feed's decoded audio and after our microphone's mute (−38 dBFS peak); SIP participants from conference-audio-levels (rx_peak > 25). Held 0.6 s after the last loud level.
- **Video loss,** from sylk-desktop and Sylk Mobile: each feed's received video bitrate, sampled once a second, averaged (EMA α 0.4). Below 80 kbit/s after a 5 s warm-up the tile is *degraded*: it keeps its place and shows the initial. Degraded for 30 s it is *demoted* to the carousel; back above 80 kbit/s it recovers, a demoted one returns to the grid. A tile the user placed (Replace) or a speaker is never demoted. No video bytes at all for 20 s asks the server once for the video again (videoroom-session-update video false, then true 300 ms later: Janus sends a keyframe).
- **Pausing:** 2 s after the window is minimized or hidden every feed's video is paused (videoroom-session-update video false), audio going on; resumed when it is visible again; a feed attached meanwhile starts paused. The video loss check stops meanwhile.

4.7 Notifications

Name	Data
VideoroomSessionDidChangeState	state, reason, room
VideoroomSessionAudioMutedDidChange	muted, room
VideoroomSessionGotMessage	type, content, content_type, sender_uri, sender_name, timestamp
VideoroomSessionDidDeliverMessage	message_id, delivered, code, reason

4.8 Several video feeds per participant (extravideo.py)

A participant publishes one audio and any number of videos (at most 4 extra, server limit). Each extra video is a publisher of its own, joined over the room's existing sylkRTC connection with videoroom-join + parent (our publisher's session id) + label ("Living", "Camera 2"), audio false. The gateway (feature multiple-video-feeds):

- names it "<owner's display name> ", gives it no MSRP chat leg (so it is not another participant of the conference), lists it in publishers with parent and label;
- refuses a parent that is not a publisher of the same room, or a fifth extra video;
- ends the children with their parent.

Sources: a camera of the computer (GStreamer device monitor, v4l2 preferred), or an RTSP camera (rtspsrc, TCP, user/password; video stream only: select-stream drops audio and metadata, any other pad goes to a fakesink, an unlinked pad stops rtspsrc). Pipeline, complete before the offer so the offer carries the SSRC:

```
source → decodebin → convert/scale(add borders)/rate(10 fps) → capsfilter WxH → tee → queue
                                     ↳ queue → convert → appsink (its ti
```

- The codec is the one the answer chose for our own publisher (H264 here).
- **Codec preferences narrowed** after the offer (and again after the answer) to encoding-name, payload, clock-rate and SSRC: the offer's fmt (profile-level-id=42e01f) does not match what x264's constrained baseline produces (42c0..) and webrtcbin refuses the packets (not-negotiated, seen as rtspsrc's "Internal data stream error" ~1 s in). Our own publisher already did this.
- Resolutions (16:9, 10 fps): **640×360 / 500 kbit/s (default)**, 1280×720 / 1200 kbit/s, 320×180 / 200 kbit/s; changed while sending by setting the capsfilter caps and the encoder's bitrate (no renegotiation).
- Our own extra videos are tiles here from their own appsink; the room's publishers event for them is skipped (their id is ours, or their parent is our session).
- Statistics: request_stats() asks every extra video's webrtcbin too.
- RTSP cameras that worked are remembered with name, address, user, password and resolution (ui.conf videoroom/video_sources, JSON, newest first, 20).
- Speakers: an extra video can be a speaker (videoroom-configure takes its id); our own are looked up in session.extra_videos, not among the participants.

Network cameras (camerafinder.py): ONVIF WS-Discovery (a Probe for dn:NetworkVideoTransmitter to 239.255.255.250:3702, 3 s; name and hardware from the onvif://www.onvif.org/name|hardware/scopes), the RTSP address with ONVIF SOAP GetCapabilities → GetProfiles → GetStreamUri (WS-Security UsernameToken digest), and Bonjour_rtp._tcp when python3-zeroconf is installed. test_stream() pulls the first picture (rtspsrc → decodebin → appsink, 12 s).

5. User interface

5.1 Join Conference panel (sessions.ConferenceDialog)

- Media line: **Chat** (always checked, not changeable), **Audio**, **Video**. Video is shown only when the account's domain runs SylkServer and the media can be done here. **Video starts unchecked** every time the panel opens.

- **Video** checked: audio and video over WebRTC (Audio unchecked: the microphone starts muted), chat over MSRP. **Audio** only: SIP audio and chat to the bridge, as before.
- The room field takes a room name, room@host or a link https://host/conference/room. Rooms of another domain are looked up while typing, and **Join waits for the answer** (§6.2). It opens with **the room joined last**; a new random room only when no conference was joined yet.
- History: the rooms joined, most recent first (ui.conf conference_dialog/recent_rooms, 20), rebuilt each time the panel opens. Every outgoing session to a conference address (this panel, a call or chat from elsewhere, an escalation) puts its room first; a room of the account's own conference server is kept by its name alone, and 1234, 1234@conference.<domain> and 1234@video-conference.<domain> are one room.
- **Publish** (shown with Video checked): the videos we send, one line each, checkable. With a gateway that announces multiple-video-feeds (asked when the panel opens, §3), three groups with a gap above each title:
 - *Local cameras*: the first checked one is our publisher's camera (Blink's video device is set to it), the others go as extra feeds;
 - *Screens*: one entry per monitor ("My screen 2: DP-1, 2560x1440"), at most one checked: shared as soon as the room is joined (we join with the camera off until then); every checked camera is then an extra feed, started once the screen has taken the camera's place (the device is free then), at most 30 s later;
 - *RTSP cameras*: the remembered ones; **Add network camera...**, Forget; right click: Send at (resolution), Rename, Forget; double click renames.

Without that feature: cameras and screens only, one choice. On opening: an **external (plugged in) camera** is checked (Linux: the USB port's removable attribute in sysfs, else built-in names such as "Integrated", "FaceTime"), else the camera in use; no screen, no RTSP camera. **One camera or screen always stays checked** with Video. **Changing the room unchecks the RTSP cameras** (a profile's room keeps its own). Extra feeds are named by their label, else "Camera N".

- **Add network camera** (camerasetup.py): the cameras found on the network (ONVIF, Bonjour) and those tested before (bold, ✓, under the name given, first; "not found now" when they do not answer), filter and sort (name/address); picking one fills name, address and the user and password used before with that host; **Test** pulls a picture (an ONVIF camera's RTSP address is asked first) and shows it; **Add** only after a test of what is in the fields; Send at; renaming a tested camera saves and refreshes the list. 520-720 px wide.
- **Profiles** (left of Join): a saved composition: room, Audio, Video, the checked video lines (keys camera:<device>, screen:<device>, rtsp:<url>) and the invitees (address and name), in ui.conf conference_dialog/profiles (JSON, by name). Picking one fills the panel; Save... (a new name, or the chosen one to change it; replacing another asks); Delete asks. The panel opens with no profile chosen.
- Heights: the video lines get twice the stretch of the invitees; without Video the invitees take it all.
- Bottom line: "Conferencing provided by SylkServer at "; without SylkServer the original MSRP switch (RFC 7701) note; with missing packages, the sudo apt install ... line.
- Invitees: **with Video, invited through the video room** (videoroom-invite once joined: a WebRTC client gets the video conference, a SIP address a call with audio and chat, and invite-status reports each step); without, REFERred through the MSRP session, its outcome logged (SIPConferenceGotAddParticipantProgress / DidAddParticipant / DidNotAddParticipant).

A Video Call to a room contact joins the same way (WebRTC + MSRP chat). An audio call to a room whose video room is open joins with chat only.

5.2 Video conference window (videoroomwindow.py)

Grid (layout in §7): pictures cropped to fill their tile (a shared screen, ours, fitted). Our own camera is mirrored, as "Me", in the grid or floating over it as a thumbnail.

Name pills at the tiles' outer corners: top for the top row, bottom for the others, left or right by side, centred for a middle column. A pill that would be under the call bar or our floating thumbnail is moved above it.

Badges on a tile: a yellow raised hand (the corner beside the name, same edge), a green border while the participant speaks, the stats pill on hover (codec · kbit/s, top middle), the initial with "no video" while its video is lost.

Call bar (the video call window's, at 75 %, lifted above the bottom row's name pills):

Segment	
Mute	microphone
Camera (menu)	each camera (System Default, then the usable ones), the screen (one entry per monitor), Turn camera off/on; red "Camera off" while off
Stop sharing	red, only while the screen is shared
Speakers	speaker selection (\$5.6); with two or more others with video, or speakers set
Chat	the room's MSRP chat in front (started if there is none)
Raise/Lower hand	with more than two of us (or our hand still up); yellow while up
More (menu)	Share screen / Stop sharing screen (connected only), Hide myself / Show myself, Show me as a thumbnail / Put me back in the grid, Add video... (with multiple-video-feeds), Invite participants..., Conference info..., Share conference address..., Mute everyone, Lower all hands
Full screen / Window	
Leave	

It hides after 5 s without mouse movement, together with the carousel, and comes back when the mouse moves.

Carousel (above the call bar, no background): the tiles that are not in the grid: past the grid's 8, audio only participants, those demoted after losing video, and, with speakers set, everyone but the speakers. Video thumbnails 80×45 cropped, name as tooltip, small red mic when muted; audio participants as round avatars (initials on the contact colour, `blink.avatars`), the video visible around them, with a green ring while speaking and a hand badge. The wheel or the carets at its ends scroll it. **Pick and replace:** a click picks a video thumbnail (blue border), the grid's remote tiles show they can be replaced, a click on one swaps them; Esc, a second click or a click elsewhere cancels; not with speakers set.

Right-click on a tile: Mute/Unmute, Lower hand, Show as the speaker / as second speaker / Stop showing as speaker, Remove from the conference (SIP participants). On our own: Show me as a thumbnail / Put me back in the grid, Hide myself, Raise/Lower hand, the speaker items. On one of our extra videos: Send at (resolution, live), Stop this video, the speaker items; someone else's extra video has no Mute nor hand.

Our own tile: a small round button at its top right, under the mouse, moves it between the grid and the thumbnail; it takes the first corner clear of the name pill and the hand badge. The thumbnail (20 % of the width, 160–320 px, 16:9) can be dragged and snaps to the nearest corner. *Hide myself* (connected, with someone else's video to show) removes it from the window only; it comes back by itself when nobody else's video is left.

Screen sharing: More → Share screen or the Camera menu asks the desktop which screen (`blink.screen_share`, the screen is a video device); then our tile is hidden, the window minimizes,

and, with no speakers chosen, the speakers follow our videos: the screen alone, or the screen and our one other video (both speakers); with more than two videos of ours, nobody. Updated as our extra videos connect and end, until someone chooses speakers. When it stops (Stop sharing, or the desktop's own indicator) the camera comes back, the speakers go back to the grid if we set them, our tile comes back and the window is restored as it was.

Status pill: while connecting, to say why the room ended, and short notes: "X raised a hand", "Muted by X", "Calling X...", "X answered", "X: ()", "Sharing your screen". Window title: Room <name> · <n> participants · mm:ss once connected.

Contacts dragged from the contact list onto the window are invited. Listed in the main window's **Window** menu while open.

5.3 MSRP window, for a room with a video room

Control	Behaviour
Header Chat / Audio / Video	only start what is not there, or bring up what is; pressed while it lasts. Audio and Video bring up the video window when a video room is open
Header Video	joins with audio and video over WebRTC, which removes the session's SIP audio; shown for rooms that can have video, a foreign SylkServer room included even without isfocus
Header Share	the share dialog (§5.4), for rooms with a SylkServer configuration
Microphone	mutes the video room, in step with its window and a moderator
Add audio / Start audio and MSRP chat	not offered while a video room is open
Hang up	ends the chat and leaves the video room (its window closes)
Tile x	in a room: hangs up (video room too), the tile stays; disconnected: closes the tile
Input bar	enabled when the MSRP chat is connected, or when the video room is (the chat then goes over WebRTC)
Participants list	dropping contacts invites them: through the video room while we are in it, else REFER over MSRP; the audio bridge is not listed; our own row shows audio and video while the video room is up
Header Audio	hidden while the video room is up: only Chat and Video
Session tile	shows the audio and video icons while the video room is up
Opening	joining with video opens the MSRP window minimized (or leaves it as it is when in use); the video window's Chat button brings it to the front
Ending	the MSRP chat Blink opened for the video room (Join panel with Video, a video call to a room, the video window's Chat button) ends when the video room ends, however it ends; a chat that was there before (escalated to video) goes on

Control	Behaviour
Hangup tone	not played when the SIP audio ends because the room's audio moved to the video room

Room chat over WebRTC: when the MSRP chat is not connected, messages go out with videoroom-message and those of the room come in through the same tile, kept in history like MSRP ones (ChatStreamGotMessage / ChatStreamWillSendMessage with a stand-in stream); an id made of sender, time and content keeps the backlog the room replays from being stored twice. When the MSRP chat is connected it carries the chat and the WebRTC copies are ignored.

5.4 Share dialog (sharedialog.py)

As Sylk Mobile's ShareConferenceLinkModal, from the configuration of the room's own domain:

- Web: <publicUrl>/conference/<room>
- SIP: <room>@<conference domain> (the MSRP switch address)
- PSTN access number and room number, when conference.pstnBridge is set
- QR code of sylk://<domain>/conference/<room> (python3-qrcode or segno)

Each address can be selected and copied; Copy puts all of them on the clipboard, E-mail... opens a message with them.

5.5 Incoming invitation

An INVITE from a conference focus that carries Referred-By (the focus copies it from the REFER) shows as an invitation from that person: title "Conference Invitation", their name and picture from the contacts (else their address), the note "invites you to conference ". Accepting is as before: the session is the room's. A WebRTC user's invite reaches SIP addresses with audio and chat (media="audio,chat" on the REFER), phone numbers with audio only.

5.6 Speaker selection (speakerdialog.py)

As Sylk Mobile's SpeakerSelectionModal: Grid / 1 speaker / 2 speakers; one list per slot of the participants with video, ours as "Me", sorted by name; a click picks, a click on the picked one clears it, one person in one slot only. Apply sends videoroom-configure; the window changes when the room's configure event comes (whoever set it). One speaker fills the window, two side by side, everyone else in the carousel.

5.7 Add video (in the room)

More → Add video...: name, a camera of this computer or an RTSP camera (used before, or **Find on network...**, which opens Add network camera), address, user, password, Send at. The used-before list shows "name (host)", the address as tooltip, and does not widen the dialog.

5.8 Conference info (infopanel.py)

More → Conference info...: an ordinary top-level window (not a dialog: window managers keep a parentless dialog above every window of the application), 420 px wide, as tall as the room's window, placed beside it on the side the screen has room for (at the screen's edge when neither side has). Listed in the main window's Window menu under its room while open; closes with the room's window. Refreshed every 2 s, rates from the difference of the byte counters.

- Top: Room, State, Sending (codec, encoder, SSRC), Server (our video's loss and RTT from remote-inbound-rtp), **Upload ↑** and **Download ↓** totals (sum of every rate: our audio, video and extra videos up; every feed and the SIP mix down).

- A tree: Me, then our extra videos (“Living (mine)”: source, size, fps, kbit/s sent, loss), then the others, then “SIP callers, mixed”. Each entry: a status (state, ↑/↓ kbit/s), and under it Address, Audio, Video, Video received/sent, Jitter, Shown (grid, carousel, hidden, speaker, hand up, video lost, placed). Entries are updated in place, keeping their expansion.

6. Connecting to foreign domains

6.1 What a foreign room is

A room whose host is neither the account’s conference server nor in its domain, given as `room@conference.example.com`, `room@example.com` or a link `https://example.com/conference/room`. Its SylkServer is found by the same discovery as for accounts, for the **room’s** domain: `conference.` and `videoconference.` are stripped from the host and `_sylkserver.<domain>` is looked up.

Example, `https://domain.sylk.link/conference/others4`:

From	Value
<code>TXT _sylkserver.domain.sylk.link</code>	<code>https://domain.sylk.link/sylk-config.json</code>
<code>wsServer</code>	<code>wss://domain.sylk.link/ws</code>
<code>defaultConferenceDomain</code>	<code>videoconference.domain.sylk.link → video room oth-</code>
<code>MSRP switch (convention)</code>	<code>ers4@videoconference.domain.sylk.link others4@conference.domain.sylk.link (videoconference. → conference.)</code>
<code>iceServers</code>	<code>that server’s (here STUN only)</code>
<code>defaultGuestDomain</code>	<code>guest.domain.sylk.link</code>

`RoomTarget (__init__.py)` holds this; `remote` is true when its `wsServer` is not the account’s own.

6.2 Discovery while typing

`resolve_room()` uses the cached configuration, or discovers it (up to 12 s, `DISCOVERY_TIMEOUT`). The Join Conference panel starts the lookup 0.4 s after the typing stops, says “Looking up the SylkServer of ...”, keeps **Join disabled** until the answer, then shows “Conferencing provided by SylkServer at conference.” or “No SylkServer found for : the room is joined over SIP”. Join uses the answer it got.

6.3 Identity

The gateway authenticates nothing on account-add and videoroom-join: it checks the domain against its `sip_domains` (default *) and the room’s allow/deny policy. Blink joins as **its own address** first; when the server answers SIP domain not allowed it joins once more as a **guest** of that server: `<display name><4 hex>@<defaultGuestDomain>`, with a throwaway HA1, the display name kept.

6.4 Chat with a foreign room

The MSRP chat goes over SIP to `room@conference.<domain>`, through our own proxy and the other domain’s. When it cannot connect (a proxy refusing to relay, an MSRP port that cannot be reached) the chat goes over the video room’s WebRTC connection instead (§5.3).

6.5 What the foreign server needs

Found while making `domain.sylk.link` work (sylk-suite in Docker inside an LXC):

Problem	Symptom in Blink	Fix
Its OpenSIPS refused INVITEs from other domains to its conference domain	403 Relaying denied	opensips-config-sylkserver: CONFERENCE_SERVER_PUBLIC (default 1) lets requests to CONFERENCE_SERVER_PREFIX.<local domain> through the relay check
SylkServer advertised its container address in the MSRP path	media stream timed-out while starting; [msrp] ... advertised the private address 172.18.0.10:...	[MSRP] msrp_hostname set to the public name (sylk-suite endpoint: FULL_DOMAIN)
SylkServer listened on a random port per session	same, once the host was right	sylkserver: [MSRP] port (default 2856), one listening socket for all incoming sessions, each connection given to its session by the To-Path session id; published by Docker and forwarded by the LXC host (lxc config device add ... proxy listen=tcp:<public ip>:2856 connect=tcp:0.0.0.0:2856 nat=true)
The self-signed certificate of the SylkServer package	peer certificate signer not found ... for CN Sylkserver issued by AG Projects Development	sylk-suite: the certbot certificate mounted into the container, joined into the one file SylkServer takes and set as certificate
The certbot certificate had expired (renewals never ran)	... EXPIRED in SylkServer's log	the certbot container recreated with the fixed command, renewed by hand
Its OpenSIPS rebuilt the Contact of INVITE replies from the URI only	no isfocus: no conference-info, no participants list	opensips-config-sylkserver: the Contact is rewritten only to strip pn-* URI parameters, keeping its header parameters (isfocus)
sylkserver from source needs python3-sipsimple >= 5.6.0	—	image rebuilt with --no-cache (5.6.4 from the repository)

Blink explains MSRP failures in the Activity and MSRP logs ([msrp], NOTE): a private address in the path, no answer within the timeout, or a certificate refused, with the address and the certificate's subject and issuer.

6.6 Our side

Calling a foreign room's MSRP switch also needs **our** proxy to send the INVITE out for a local user; 403 Relaying denied from our own proxy is a policy there, not in Blink.

7. Tile layout

In the grid	Layout
1	the whole window
2	side by side; one over the other when the window is at most as wide as high (TWO_IN_A_COLUMN_ASPECT 1.0)
3	3×1 when width/height ≥ 2.2, 1×3 when ≤ 0.75, else 2 over 1
4–8	two rows, the longer on top, a shorter bottom row centred: 2×2, 3+2, 3×2, 4+3, 4×2

- **Our own video** floats as the thumbnail when that leaves an even grid of the others (3 of us: 2×1 and the thumbnail, 5: 2×2, 7: 3×2, 9: 4×2), or when the others fill the grid; with an even number of us it is in the grid. The tile's button / menu overrides this (floating with one other or more, or in the grid).
- **At most 8** in the grid; the others, audio only participants and demoted ones go to the carousel (§5.2). Speakers set: the speakers alone (one, or two side by side).
- Tiles take the whole cell, pictures cropped to fill. Ours first, the others in joining order.

8. Logging

Tag	What
[webrtc] [sylkrct] [videoroom]	availability, the lookup of a foreign domain connections, accounts added, failed requests join, offer and answer (summary; full SDP at debug level), codecs, participants, ICE states, GStreamer errors, mute, camera
[conference] [msrp] [screen sharing]	joins, foreign room addresses, invitations why an MSRP stream did not start asking the desktop, the screen given, sharing stopped from the desktop, the camera back

[videoroom] also has: speakers set (and by whom), hands raised, muted by someone, invite progress, video lost (N kbit/s) / moved to the carousel / video back, pausing and resuming the feeds, sending the screen at ... / sending the camera at

9. Limits and open items

- Outgoing camera video is 640×480 at 24 fps, letterboxed; the screen 1280×720 at 15 fps. 1080p would need H264 level 4.0, more than the rooms negotiate.
- Audio uses the system's default devices, not Blink's audio device settings.
- No composing indications or file transfers over WebRTC; the room's file-sharing event is not shown; no conference recording.
- Pausing the feeds follows minimized/hidden only: Qt cannot tell a covered window on Linux.
- The speaking thresholds (−38 dBFS, 25 on the mixer's scale) and the video loss threshold (80 kbit/s) are untested guesses; a nearly still VP9/H264 picture can dip under 80 kbit/s.
- SylkServer's shared MSRP port keeps one stale entry when an outgoing actpass session is switched to a connector by sipsimple.
- After a certbot renewal SylkServer takes the new certificate only when restarted.
- Extra videos: 10 fps fixed; a local camera opened by GStreamer for an extra video must not be the SDK's (the one our publisher sends); one screen at a time (the SDK's video device).
- ONVIF discovery sends one multicast probe on the default interface only; Bonjour needs python3-zeroconf.

10. Changes since the first version (2026-10-24), for the Cocoa port

Each item says what Blink Cocoa needs; *Qt only* marks what comes from GStreamer or Qt and has no counterpart with libwebrtc. Server and SDK changes are listed last: they are deployed once and serve both clients.

10.1 Window and layout

1. **Layout by count** (§7): 1 full; 2 side by side or stacked (portrait); 3 as 3×1 / 1×3 / 2 over 1 by the window's shape; 4-8 two rows; at most 8. Cocoa: port the rules and the three aspect thresholds.
2. **Our own video as a floating thumbnail** when that keeps the grid even; draggable, snapping to corners; a button on our tile and menu items to force it either way; *Hide myself / Show myself* (only when connected with someone else's video).
3. **Pictures cropped to fill** their tile (screen: fitted).
4. **Name pills at the outer corners**, moved above the call bar and the thumbnail when under them; the call bar at 75 % and lifted above the bottom row's pills.
5. **Carousel** for what is not in the grid: video thumbnails 80×45, audio participants as round avatars with no background, carets/wheel, hidden with the call bar; **pick and replace**.
6. **Audio only participants** (SIP callers, feeds whose offer has no video) never in the grid.
7. **Contact names** for participants (contacts list first).
8. **Full screen** button reads **Window** while in full screen.

10.2 Controls

9. **Camera** segment is a menu: cameras, the screen, turn off/on.
10. **More** menu: Share screen, Hide myself, thumbnail/grid, Invite participants..., Share conference address... (Share moved here from the bar), Mute everyone, Lower all hands.
11. **Raise hand** button (more than two of us), hand badges, "X raised a hand".
12. **Speakers** button and dialog (§5.6); speakers layout.
13. **Chat** button: the MSRP window in front, starting the chat if needed.
14. **Right-click menus** on tiles (§5.2).
15. **Invite** from the room (contact picker, or contacts dropped on the window) and invite-status notes.

10.3 Media behaviour

16. **Speaking** highlight (green border / ring). Cocoa: from libwebrtc's `audioLevel` in the inbound-rtp stats for WebRTC feeds, `conference-audio-levels` for SIP participants.
17. **Video loss** handling (§4.6): degrade, demote after 30 s, keyframe nudge after 20 s without bytes, recovery. Cocoa: the bitrate from `bytesReceived` in the stats.
18. **Pausing feeds** out of sight: Cocoa already has it (occlusion); unchanged.
19. **Stats pill** on hover (codec, kbit/s).
20. **Screen sharing**: the screen as the camera input, 720p 15 fps 1800 kbit/s; our tile hidden, window minimized and restored; single speaker while sharing when no speakers were chosen; Stop sharing button; stopping from the OS. Cocoa: the screen capture device of patch 32 (macOS) the same way.
21. *Qt only*: **video SSRC in the offer** (GStreamer offered none for a chain built after the answer: Janus dropped our video); **I420 frames** taken from the screen device; the camera input buffer sized for a 4K frame; **depayloaders without header extensions** (GStreamer abort after pause/resume).

10.4 MSRP window and calls

22. **MSRP window minimized** when joining with video; brought up by the video window's Chat.
23. **Header Audio hidden** while the video room is up (Chat and Video only).

- 24. **Session tile and our participant row** show audio and video while the video room is up.
- 25. **The chat opened for the video room ends with it** (Leave, window closed, or the server ending it); a chat that existed before stays.
- 26. **No hangup tone** when the SIP audio ends because the audio moved to the video room.
- 27. **Incoming invitation panel** shows the inviter from Referred-By and “invites you to conference X” (§5.5).
- 28. **Our MSRP leg identified as us:** +sip.instance (the SDK’s instance_id) on the MSRP INVITE to a room, and sip_instance on videoroom-join. Cocoa: needed too, or the other clients see the Cocoa user twice and, after leaving the video room, as a SIP caller.

10.5 Server (sylkserver)

- 29. webrtcgateway: videoroom-join takes sip_instance; that endpoint of the focus is the device’s own (not a surrogate SIP participant). Older gateways ignore the field.
- 30. webrtcgateway: the REFER for a WebRTC user’s invite carries media="audio,chat" for SIP addresses, media=audio for phone numbers (PSTN gateways choke on an unknown m= line); the conference parses the quoted list.
- 31. conference: an audio stream that times out ends the session even when it also has chat (chat through an MSRP relay outlives a dead client).

10.6 SDK (python3-sipsimple)

- 32. The RTP timeout never fired under Python 3 (the transport’s direction is bytes): fixed; calls whose incoming RTP stops for rtp.timeout now end.
- 33. Linux screen capture (patch 52): the portal session’s Closed signal ends the capture, as the desktop’s sharing indicator does it. macOS has its own capture (patch 32).

11. Changes of 2026-10-26, for the Cocoa port

Numbered on from §10. The gateway, its protocol and the behaviour are the same for both clients; *Qt only* marks GStreamer details.

11.1 Server (sylkserver webrtcgateway)

- 34. **ready-event features:** ["sip-instance", "multiple-video-feeds"] (GATEWAY_FEATURES, sylkrtc.ReadyEvent.features, optional). Old clients ignore it (sylkrtc.js only checks sylkrtc). Cocoa: read it; offer extra feeds only with multiple-video-feeds.
- 35. **Child publishers:** videoroom-join parent + label (§4.8): video only, no chat leg, named “ ”, validated (same room, a top-level publisher, at most 4), ended with the parent, listed in publishers with parent/label. Old clients show them as separate video-only participants with that name. Against an old gateway the fields are ignored: each extra feed would be another participant with its own chat leg; hence the feature check.
- 36. Speakers (videoroom-configure active_participants) accept a child’s id.

11.2 Publishing several videos

- 37. **Extra video feeds** (§4.8): a second local camera, the screen together with cameras, RTSP cameras. Each is a sendonly peer connection of its own joined with parent/label; its tile here is drawn from the local source. Cocoa: an RTCPeerConnection per extra feed with an RTCVideoSource fed from AVFoundation (second camera) or an RTSP decoder; join with parent = our publisher’s session id.
- 38. **Resolutions** 640×360 (default, 500 kbit/s), 1280×720 (1200), 320×180 (200), 10 fps; changeable while sending; remembered per RTSP camera.
- 39. *Qt only:* narrow codec preferences to codec + SSRC after the offer (fmtfp mismatch → not-negotiated); rtspsrc select-stream video only (Hikvision sends audio: an unlinked pad stops it); GStreamer errors logged in full.

- 40. **Auto speakers while sharing the screen:** screen alone → us; screen + one more video of ours → both; more than two → none. Follows our feeds until someone chooses speakers.
- 41. Our own extra videos can be **speakers** here too (looked up in our extra videos).

11.3 Join Conference panel (§5.1)

- 42. **Video sources** list with Local cameras / Screens / RTSP cameras groups (with multiple-video-feeds; one choice without); spacing above group titles.
- 43. Defaults on opening: Video unchecked; external camera checked; no screen; no RTSP camera; always one camera or screen checked; a room change unchecks RTSP cameras.
- 44. **Add network camera** window: ONVIF WS-Discovery + GetStreamUri (WS-Security digest), Bonjour _rtsp._tcp; tested cameras first under their names; filter, sort; Test before Add with a preview; credentials remembered per camera.
- 45. **Profiles:** room, media, video lines, invitees; save/load/delete; none chosen on opening.
- 46. **Last room** in the field on opening, random only with no history; any dialed conference recorded; room forms deduplicated.
- 47. **Invitees through the video room** when joining with video (videoroom-invite, status reported); dropping contacts on a room's participants does the same while in its video room; REFER outcomes logged otherwise.
- 48. The checked videos are read before the room moves to the top of the history (moving it changed the field and unchecked the RTSP cameras: a bug of the room-change rule).
- 49. Heights: video lines twice the invitees; invitees take the space without Video.

11.4 Room window and Conference info

- 50. **Conference info** window (§5.8): narrow tree beside the room's window, totals up/down, our extra videos under us, an ordinary window listed in the Window menu.
- 51. **Add video...** (More, with the feature) and the tile menu's **Send at / Stop this video** for our extra videos (§5.7).
- 52. Statistics of our extra videos (outbound-rtp, remote-inbound-rtp) in Conference info.