AUTHORIZATION

BASE URL=https://api.eyeson.team

To create a room and register webhooks (for any rooms), you must provide authorization by sending the API key in the HTTP header.

ROOM

Create room

Initialize room with <api_key>. Response contains <access_key> and <guest_token>.

POST /rooms

Authorization: <api_key>, required: user[name], options[sfu_mode]=disabled, options[widescreen]=true

Room parameters

id, name, user[id], user[name], user[avatar], options[show_names], options[show_label], options[exit_url], options[recording_available], options[broadcast_available], options[reaction_available], options[layout_available], options[guest_token_available], options[lock_available], options[kick_available], options[sfu_mode], options[widescreen], options[background_color], options[audio_insert], options[audio_insert_position][x], options[audio_insert_position][y], options[custom_fields][locale], options[custom_fields][logo], options[custom_fields][hide_chat], options[custom_fields][virtual_background], options[custom_fields][virtual_background_allow_guest], options[custom_fields][virtual_background_image]

Lock room

Get room details

POST /rooms/<access_key>/lock

GET /rooms/<access_key>

Stop room

Get list of running rooms

DELETE /rooms/<access_key>

GET /rooms

Force stop room

DELETE /rooms/<room id>

USERS

Get user details

GET /rooms/<access_key>/users/<user_id>

Join

From API-Response: links[GUI] https://app.eyeson.team/?ACCESS_KEY

Register user

POST /rooms

REQUIRED id, user[name], RECOMMENDED user[id]

Register guest

POST /quests/<quest token>

Kick

DELETE /rooms/<access_key>/users/<user_id>

LAYERS

PNG / WEBP - 1280x960 (Default) or 1280x720 (Widescreen)

Add laver

POST /rooms/<access_key>/layers

Image from web url as overlay

```
curl -X POST \
  -d "url=https://www.domain.com/file.webp" \
 -d "z-index=1" \
  "https://api.eyeson.team/rooms/$ACCESS_KEY/layers"
```

Local image in background

```
curl -X POST \
 -F "file=@path/to/local/file.png" \
 -F ..z-index=-1" \
  "https://api.eyeson.team/rooms/$ACCESS_KEY/layers"
```

Clear layer

DELETE /rooms/<access_key>/layers/<layer_index>

LAYOUT

Apply layout

POST /rooms/<access_key>/layout

REQUIRED at least one users[]="

VALID names: one, two, four, six, nine, present-lower-3, present-upper-6, present-two-upper-6, present-upper-right-9, present-vertical-9

Custom layout

Layouts can contain multiple spots, user spots are filled in order and can overlap but not reach outside the display.

```
POST /rooms/<access_key>/layout
map = [[x,y,width, height,(object_fit)],[...]]"
```

object_fit:

cover: Scales content to fill spot, exceeding boundaries if needed contain: fit content into spot - letterboxed auto: narrow videos get contain

Position users

Their placement in the user list is what matters

```
users[] = " ";
users[] = $USER_ID;
users[] = " ";
users[] = " ";
```

Freeze position

Empty spots won't get filled

layout = custom

Voice activation

Users with active microphones rank higher in the user list

PLAYBACK

Only webm files can be looped. Optimal conversion via ffmpeg:

```
ffmpeg -i input.mp4 -r 25 -g 50 -c:v libvpx -b:v 5M -c:a
libvorbis output.webm
```

Start

POST /rooms/<access_key>/playbacks

Stop

DELETE /rooms/<access_key>/playbacks/<play_id>

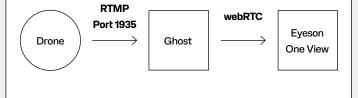
eyeson API CHEATSHEET

ADD DRONE OR RTMP/RTSP SOURCE

- Download Go Host Streaming Client (Ghost) https://github.com/eyeson-team/ghost/releases
- Set drone to stream to IP of Ghost machine (use ngrok if necessary), RTMP, port 1935

./rtmp-server_<0S_VERSION>.exe --user Drone --delay 60
https://app.eyeson.team/?guest=[\$GUEST_TOKEN]

3. Drone joins call



PERMALINK

Create a persistent link for a call

Create permalink

POST /permalink
HEADERS Authorization

Delete permalink

DELETE /permalink
HEADERS Authorization

Delete host user from permalink

DELETE /permalink/<permalink_id>/users/<user-token>

Start meeting from permalink

POST /permalink/<user_token>

Add host user to meeting

Host users can start a meeting, only give away guest links.

POST /permalink/<permalink_id>/users

SNAPSHOTS & RECORDINGS

You will need to have the id to retrieve a recording or a snapshot. You can list all the recordings and snapshots in regular intervals. Alternatively, use the observer.

Start / Stop recording

POST /rooms/<access_key>/recording
DELETE /rooms/<access_key>/recording

Retrieve recording

GET /recordings/<recording_id>

Get list of recordings

GET /rooms/<room_id>/recordings

Create snapshot

POST /rooms/<access_key>/snapshot

Retrieve snapshot

GET /rooms/<access_key>/snapshots/<snapshot_id>

Get list of snapshots

GET /rooms/<room_id>/snapshots

OBSERVER

Get all meta data of the call by using the Observer: current participants, layouts, recording and snapshots handling.

https://api.eyeson.team/rt?room_id=<room_id>
(can be wss:// protocol in some cases)
api_key=<YOUR_API_KEY>
Subscribe to RoomChannel

Event types

room_update
participant_update
recording_update
snapshots_update

FORWARD STREAM

The API URL must contain the ROOM_ID in contrast to ACCESS_KEY in other API calls. forward_id MUST be unique for each forward!

Forward source

POST /rooms/<ROOM_ID>/forward/source

Forward MCU

POST /rooms/<ROOM_ID>/forward/mcu

Forward playback

POST /rooms/<ROOM_ID>/forward/playback

End forward

DELETE /rooms/<ROOM_ID>/forward/<FORWARD_ID>

NOTES	