

AUTOMATIC MODE

- multiple guests join one OBSN chat room
- by default guests can hear and see each other in the chat room at reduced quality/bitrate
- the director can simultaneously pull higher-quality video/audio from each of the guests to use in a production
- in OBS on Windows, this can be done by adding an OBS browser source using a solo link for a specific guest
- in OBS on Mac you need a workaround, see **#WIN VS. MAC**

DIRECTORLESS MODE

- mix and match &view and &push links to create manual mode rooms
- example with three parties:
 - obs.ninja/?push=p11&view=p2,p3
 - obs.ninja/?push=p2&view=p1,p2
 - obs.ninja/?push=p3&view=p1,p2
- this mode does not create scenes, you will have to do this yourself, for example inside OBS

GENERAL

- OBS.Ninja (OBSN) sends video and audio between web browsers anywhere in the world
- latency is typically very low, often under 200ms one way, allowing for fluid interaction
- OBSN and OBS are two entirely different things, but you can use OBSN inside OBS
- OBSN can be used stand-alone or with other video mixers, OBS is optional
- connections are point to point, from browser to browser, encrypted
- a handshake server is used to facilitate connections
- on desktop, use Chrome or Firefox, OBSN will not work in desktop Safari
- on iPhone, use Safari, see **# IPHONE**
- OBSN uses a chain of URL parameters: &<parameter>=<value>&<parameter>..., example:
obs.ninja/?push=myStreamID&stereo=1
- some parameters go on the sender's URL, some on the receiver's URL, some on both
- OBSN works in different modes: Automatic Room Mode, Manual Room Mode, Single Stream Mode
- avoid Wifi, the majority of connection instability comes from wireless connections

GUESTS

- obs.ninja/?room=<room name>
- stream IDs will be automatically generated
- to create reusable links, add a different &push=<streamID> for each guest



DIRECTOR

- obs.ninja/?director=<room name>
- opens the director control panel

SIMPLE EXAMPLES

Simple stream: obs.ninja/?push=stream001 | obs.ninja/?view=stream001
Simple two-way: obs.ninja/?push=s001&view=s002 | obs.ninja/?view=s001&push=s002
Receive high quality: obs.ninja/?view=s001&s&vb=10000&buffer
...

WIN VS. MAC

- getting an OBSN stream into OBS on Windows is as easy as dragging an OBSN receive link onto OBS
- on Mac display capture the **# ELECTRON APP** and send audio to a virtual device
- a/v sync compensate in OBS or via OBSN &sync parameter

SCENE

- obs.ninja/?room=<room name>&scene=0 creates an automatic scene with all guest videos at good quality in one window
- audio goes to the default audio device
- audio volume of individual participants can be adjusted by the director
- to send the sound of this scene to a specific audio device, see **# ELECTRON APP**
- if you specify &scene=1 instead of 0, you will get an empty window that the director can add participants to from the control panel

AUDIO

- Opus codec provides excellent quality at very low latency
- when using &stereo mode, the bitrate is set to 256kbit/s
- this is very generous, Opus codec sounds excellent starting at 32kbit/s
- 64 to 128 kbit/s are considered studio/broadcast quality audio

Parameter reference: params.obs.ninja
Github: github.obs.ninja
Discord: discord.obs.ninja
Reddit: reddit.com/r/obsninja

STREAM QUALITY

- connection quality and bitrates are automatically managed by OBSN, you can set your own values
- &videobitrate=<kbit/s> and/or &audiobitrate=<kbit/s>
- to specify audio bitrate, use &stereo mode on both sides
- echo cancellation, automatic gain control and noise reduction are turned off in &stereo mode, audio bitrate used is 256kbit/s
- OBSN uses default codecs: vp8 for video, Opus for audio
- &codec=vp9 enables high-quality video at the low bitrates
- vp9 can be cpu-intensive for the sending system and as a result of that might lead to lower frame rates
- &codec=h264 enables the older and less efficient h.264 codec, but it's more CPU friendly due to often being hardware-supported

IPHONE

- only allows for three simultaneous outgoing WebRTC video streams
- iPhone guests should close all Safari tabs before connecting
- iOS 14 or above supports vp9 codec (at high battery usage) if you enable WebRTC vp9 codec in Settings > Safari > Advanced > Experimental Features

PROTECTION

- by default a receiver can connect to a sender as long as they know the streamID
- you can prevent that by adding &password to the sending side or to both sides
- add &secure to the sending side to disconnect from the handshake server after the connection was established

ELECTRON APP

- lets you receive OBSN streams into a frameless encapsulated Chrome browser
- lets you send OBSN audio to an audio device of your choice, e.g. a virtual audio cable, Loopback, Blackhole or similar
- supports always on top
- comes in two flavors:
 - A) obs.ninja/electron (in Chrome tab)
 - B) full Mac/Win app (download on Github)