AUTOMATIC MODE

- multiple guests join one OBSN chat room
- by default guests hear and see each other in the chat room at reduced quality/bitrate
- the director pulls higher-quality video/audio from each of the guests to use in a production (using a solo link)
- in OBS on Windows, add an OBS browser source with the solo link for each guest
- in OBS on Mac there is a workaround to account for known browser source issues, 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=p1&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>&<paramter>..., 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 for each guest
- create a reusable link by adding &push=<uniqeStreamID> for each guest

NINJA

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
...
```

DIRECTOR

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

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

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 mixed sound of this scene to a specific audio device, see # **ELECTRON APP**
- &scene=1 instead of 0 will create an empty window that the director can add participants to from the control panel

AUDIO

- the Opus codec provides excellent quality at very low latency
- in &stereo mode, the bitrate used it 256kbit/s
- 256 kbit/s is very generous, Opus sounds excellent as low as 32kbit/s
- 64 to 128 kbit/s are considered studio/broadcast quality audio

WIN VS. MAC

- getting an OBSN stream into OBS on Windows is as eay 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

STREAM QUALITY

- audio/video quality and bitrates are managed by OBSN
- set your own values by using &videobitrate=<kbit/s> &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, default audio bitrate in this mode is 256kbit/s
- OBSN default codecs: vp8 for video, Opus for audio
- &codec=vp9 enables high-quality video at low bitrates
- vp9 can be cpu intensive for the sending system and as a result might lead to lower frame rates
- &codec=h264 enables the older and less good looking h.264 codec, which in turn is more CPU friendly as it's often hardware-supported

ELECTRON APP

- receives 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) Mac/Win app (download on Github)

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 any sender as long as they know the streamID
- you can prevent this 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 has been established