

WESC Integration Guide

Whiteley Events Service Cue (WESC)

Version 1.7.212 — June 2026

Whiteley Events Ltd · Decibel House, 32 Thyme Avenue, Whiteley, PO15 7NA

1. Introduction

This guide covers the **external integrations** exposed by Whiteley Events Service Cue (WESC) — the protocols, sockets, ports, and cue commands you need to wire WESC into a production rig alongside ProPresenter, vMix, an ATEM, a ChamSys console, an Elgato Stream Deck, an NDI fabric, and the rest of a typical AV / live events stack.

For day-to-day app usage — building runsheets, editing cues, running a show — see the WESC User Manual. This document assumes you have a working install and a runsheet open and concentrates exclusively on the surfaces where WESC meets other gear and software.

The intended audience is AV / live production technicians comfortable with networking basics, NDI, MIDI, and OSC. Where relevant we include worked examples (curl commands, Companion address strings, IAC bus setup).

WESC v1.7.212 is **macOS only**. Windows support was retired in this release; if you have an older Windows installation, run it against this version's documentation at your own risk — most integrations behave identically, but the ATEM, NDI, and Stream Deck native modules are no longer packaged or tested for Windows.

2. Network setup

2.1 NIC binding

Most production Macs have at least two network interfaces — a built-in Wi-Fi adapter and a Thunderbolt or USB-C Ethernet adapter on the show LAN. WESC's listening servers (HTTP triggers, OSC triggers, the Hollyland tally TCP server) need to bind to a specific interface so a multi-NIC machine doesn't accidentally expose the show network to a guest Wi-Fi.

Settings → **Network** → **Bind to** lets you pick which IP WESC listens on:

- **Loopback (127.0.0.1)** — default. WESC is reachable only from the same Mac. Use for OSC / HTTP triggers when Companion runs on the operator machine.
- **Specific IPv4 address** — pick the show-LAN address (e.g. 10.0.0.50). WESC's listening servers bind to that interface only.
- **All interfaces (0.0.0.0)** — listens on every NIC. Use only when you need this and have considered the security implications.

The bind address applies to:

- The HTTP trigger server (default port 7700) when **LAN mode** is enabled.
- The OSC trigger server (default port 53000).
- The Hollyland vMix-tally TCP server (default port 8099).

Loopback always remains active alongside the LAN bind — local tools and the operator UI keep working regardless.

2.2 Cohabiting with Dante and other audio-over-IP

A common rig has a **Dante** VLAN (carrying Dante Virtual Soundcard traffic) on one Ethernet adapter and a **show LAN** on another. WESC's audio path is post-OS — once you've selected an output device in **Settings** → **Audio**, the OS handles the actual transport. That means Dante audio is routed by **Dante Controller** at the OS level, not by WESC, and there is no "pin Dante to a NIC" setting inside WESC.

For control traffic (OSC, HTTP, ATEM, ProPresenter REST) bind WESC to the show LAN explicitly using **Settings** → **Network** → **Bind to** — otherwise WESC will pick whichever interface macOS hands it first, which is rarely what you want when there are two private subnets in play.

Avoid letting any WESC-bound interface auto-assign a **link-local 169.254.x.x** address. If you see one of these on your show LAN it almost always means a rogue DHCP setup or an unplugged cable; integrations like ATEM and NDI receive will appear to work locally but fail intermittently when the link-local fallback gets out-prioritised by mDNS.

2.3 Firewalls

macOS **System Settings** → **Network** → **Firewall** prompts on first run for every listening service. If you've enabled Stealth mode and/or "Block all incoming connections" you must add explicit Allow exceptions for WESC, otherwise:

- The HTTP trigger server (LAN mode) will bind but won't accept inbound connections from other machines.
- The OSC trigger server will silently drop external packets.
- The Hollyland tally server will reject the Hollyland HUB's connection attempt.

For inbound triggers (MIDI / OSC / HTTP / Stream Deck) the simplest reliable setup is: firewall on, "Block all incoming connections" off, allow signed apps to receive incoming connections.

3. ProPresenter integration

WESC drives **ProPresenter 7+** through its built-in REST API.

3.1 Configuration

Settings → **Integrations** → **ProPresenter**

Field	Default	Notes
Host	127.0.0.1	ProPresenter Mac IP. Loopback if same machine.
Port	1025	ProPresenter's default network web link port.
Enabled	on	

In ProPresenter: **Preferences** → **Network** → **Network** → **Enable Network** and confirm the Network Web Link port matches.

3.2 Supported commands

Command	Params	What it does
triggerNext	—	Advance to the next slide
triggerPrevious	—	Go to the previous slide
triggerSlide	playlistIndex, presentationIndex, slideIndex	Trigger a specific slide
triggerPlaylistItemByName	playlist, item	Find a playlist item/header by name. If the match is a header/section, trigger the first non-hidden playable item inside that section. Leave playlist blank to use ProPresenter's focused playlist.
triggerFocusedPlaylistItem	index	Trigger an item in ProPresenter's focused playlist by 0-based index
triggerPlaylistItem	playlistId, index	Trigger an item in a named/UUID/path playlist by 0-based index
triggerFocusedPresentationGroup	groupId	Trigger a group/section in ProPresenter's focused presentation
triggerActivePresentationGroup	groupId	Trigger a group/section in ProPresenter's active presentation
triggerPresentationGroup	presentationId, groupId	Trigger a group/section in a specific presentation UUID

Command	Params	What it does
clearAll	—	Clear all layers
clearSlide	—	Clear the slide layer only
triggerPlaylist	id	Trigger a playlist by UUID
triggerMedia	id	Trigger a media bin item by UUID
startTimer	id	Start a ProPresenter timer
stopTimer	id	Stop a timer
resetTimer	id	Reset a timer

3.3 Diagnostics

The **Test** button on the integration card hits GET /v1/version. A 200 response flips the connection indicator green; a refused connection means ProPresenter's network web link is off, the IP is wrong, or a firewall is in the way.

3.4 Troubleshooting

- **Connection refused** — confirm Network Web Link is enabled in ProPresenter's preferences and the port matches WESC's config.
- **Go to a playlist section such as Worship Set** — use triggerPlaylistItemByName with item=Worship Set. If Worship Set is a ProPresenter header/section, WESC fires the first non-hidden playable item under that header. Leave playlist blank to use the focused playlist, or enter the ProPresenter playlist name/path if you want WESC to resolve a specific playlist first.
- **Triggers fire but nothing happens on stage** — ProPresenter's API uses 0-based indexing for triggerSlide, triggerFocusedPlaylistItem, and triggerPlaylistItem. The first playlist item is index=0.
- **Timer commands return 404** — timer id is a UUID, not a name. Pull it from GET /v1/timers once and paste into the cue.

4. ChamSys MagicQ integration

WESC speaks two protocols to MagicQ; pick whichever is enabled on the console.

4.1 Configuration

Settings → Integrations → ChamSys MagicQ

Field	Default	Notes
Host	127.0.0.1	Console IP
Port	6553	Remote protocol; switch to 8000 for OSC
Protocol	remote	remote or osc

Enable on the console:

- **Remote:** Setup → View Settings → Network → Remote Protocol (UDP) on, port 6553.
- **OSC:** Setup → View Settings → Network → OSC In on, port 8000.

The Remote protocol is plain text over UDP and tends to be the more reliable of the two. OSC is useful when you already have an OSC bus and want one address space.

4.2 Supported commands

Command	Params	Remote message	OSC address
fireCue	playback, cue	<pb>.<cue>A	/pb/<pb>/go/<cue>
goPlayback	playback	<pb>G	/pb/<pb>/go
stopPlayback	playback	<pb>R	/pb/<pb>/release
releasePlayback	playback	<pb>R	/pb/<pb>/release

Command	Params	Remote message	OSC address
activatePlayback	playback	<pb>A	/pb/<pb>/go
setLevel	playback, level (0-100)	<pb>L<level>	/pb/<pb>/level (float 0.0-1.0)
setIntensity	channel, level (0-100 remote / 0-255 OSC)	<channel>I<level>	/intensity/<channel>

Note the level scaling differences: Remote takes integer 0-100, OSC takes a 0.0-1.0 float for setLevel and 0-255 integer for setIntensity. WESC handles the conversion internally — you always enter 0-100 in the cue editor and WESC normalises per-protocol.

4.3 Troubleshooting

- **Nothing fires** — the console silently drops malformed commands. Test with the integration card's **Test** button (sends a harmless query) to confirm packets reach the console at all.
- **Wrong playback fires** — MagicQ playbacks are 1-based; playback=1 is the first fader.
- **OSC works, Remote doesn't (or vice versa)** — verify the matching protocol is on in the console; they're independent toggles.

5. vMix integration

WESC drives vMix through its HTTP/Web Controller API.

5.1 Configuration

Settings → Integrations → vMix

Field	Default
Host	127.0.0.1
Port	8088

In vMix: **Settings** → **Web Controller** → **Enabled**, port 8088. No password unless you configure one.

5.2 Supported commands

Command	Params	vMix function
cut	—	Cut
fade	duration (ms)	Fade
transitionToInput	input, mix	Fade to input on a mix
setInputPreview	input	PreviewInput
overlayOn	overlay (1-4), input	OverlayInputNOOn
overlayOff	overlay	OverlayInputNOOff
startRecording / stopRecording	—	
startStreaming / stopStreaming	—	
playInput / pauseInput	input	
setVolume	input, volume (0-100)	SetVolume

input accepts a 1-based number, an input title, or a vMix Input Key (GUID). Names with spaces are URL-encoded automatically.

5.3 Troubleshooting

- **403 / 401** — vMix has a Web Controller password set. WESC doesn't currently support HTTP Basic auth on this integration; clear the password in vMix settings.
- **Wrong input fires** — input numbers shift when you reorder. Prefer input **names** or GUIDs in cues meant to survive a re-order.

6. Blackmagic ATEM

WESC reads Program / Preview state from a Blackmagic ATEM switcher and can optionally feed a Hollyland tally system from it. Implementation is the `atem-connection` library — pure JavaScript, no Blackmagic driver required.

6.1 Configuration

Settings → Integrations → Blackmagic ATEM

Field	Default	Notes
Enabled	off	Off by default — no traffic until you opt in
Mode	atem	atem (real switcher) or simulator
Host	10.0.0.30	Switcher IP
Port	9910	ATEM control port
Simulator port	9911	Loopback HTTP simulator
Auto-Hollyland	off	Auto-feed mapped Program/Preview channels into the Hollyland integration
Mappings	3→1, 4→2, 5→3, 6→4	ATEM input → Hollyland tally channel

Simulator mode stands up a small loopback HTTP server on port 9911 that pretends to be an ATEM. Useful for desk-checking tally mappings before you have an actual switcher in front of you. Drive it with:

```
curl -X POST http://127.0.0.1:9911/state \  
  -H 'Content-Type: application/json' \  
  -d '{"programInput": 4, "previewInput": 5}'
```

6.2 Auto-Hollyland tally feed

When **Auto-Hollyland** is on, every ATEM `stateChanged` event recomputes Program and Preview tally states and pushes them through the Hollyland integration:

- ATEM Program input → Hollyland `setProgram` (red)
- ATEM Preview input → Hollyland `setPreview` (green) — but only if it's a different channel than Program

The mapping table is the contract. Inputs that aren't in the mapping stay dark on the tally side. Adjust mappings live; the next state change picks them up.

6.3 Cue commands

Command	Params	Notes
connect	—	Force-reconnect to the switcher
disconnect	—	Drop the connection
syncHollyland	—	Push current Program/Preview to Hollyland once, even if Auto-Hollyland is off

6.4 Troubleshooting

- `atem-connection` package is not installed — the library failed to bundle. Re-run `npm install` and `electron-rebuild` from the `wesc` folder.
- **Connects then immediately disconnects** — wrong host or port. ATEMs answer on UDP 9910 only.
- **Tally not following Program** — Auto-Hollyland toggle off, or your input isn't in the mappings table.

7. Hollyland tally

WESC integrates with the **Hollyland Wireless Tally** system. The Hollyland HUB supports vMix as a switcher source; WESC implements the small subset of the vMix TCP API the HUB needs, hosting it on TCP port 8099 so the HUB

can be pointed at the WESC machine and treat WESC as if it were a vMix instance.

7.1 Configuration

Settings → Integrations → Hollyland Tally

Field	Default	Notes
Enabled	on	Server starts when enabled
Bind host	0.0.0.0	Listen on all interfaces, or pin via Settings → Network
Port	8099	vMix tally port — change only if you have a real vMix on the same Mac
Channels	16	Number of tally channels published (1-16)
Show manual buttons	on	Renders 16 manual Program / Preview / Off buttons in the integration card

On the Hollyland HUB:

- Switcher mode → vMix.
- Switcher IP → the WESC machine's show-LAN IP.
- Switcher port → 8099.

7.2 Channel state

Each tally channel is one of three states:

State	Value	Wire format	Lamp
Off	0	0	dark
Program	1	1	red
Preview	2	2	green

The wire format on every change is TALLY OK <16-character string>\r\n where each character is the state of one channel. The HUB polls / subscribes and lights its receivers accordingly.

7.3 Cue commands

Command	Params	Notes
setProgram	channel (1-16)	Set channel red
setPreview	channel	Set channel green
setOff	channel	Turn channel dark
clearAll	—	All channels off
set	channel, state (program / preview / off, or 0 / 1 / 2)	Generic

7.4 ATEM auto-feed pattern

The most reliable production pattern is:

- ATEM integration enabled with the switcher's IP.
- **Auto-Hollyland** on, mappings set so each camera's ATEM input has a tally channel.
- Hollyland HUB pointed at the WESC machine.

WESC now follows the ATEM in real time, the Hollyland HUB follows WESC, and you have no per-cue tally cues to maintain. Add explicit setProgram / setPreview cues only when you want to override the ATEM-driven state for a specific moment (e.g. light a tally on a non-camera input as a wireless cue light).

8. Stream Deck

WESC drives an **Elgato Stream Deck** directly over USB HID using `@elgato-stream-deck/node`. There is no requirement for the Elgato desktop software or Bitfocus Companion — WESC enumerates the device, paints button bitmaps, and routes presses internally.

8.1 Hardware support

Tested with Stream Deck Mk.2 (15 keys), Stream Deck XL (32 keys), Stream Deck Mini (6 keys), Stream Deck +. Keypad-only models work; the dial / touchstrip on the Stream Deck + are not yet wired.

8.2 Visual button assigner

Settings → Stream Deck

A 1:1 grid mirrors the physical device. Click a key to edit:

- **Label** — wraps to fit, painted onto the LCD with the configured colour.
- **Icon** — Unicode glyph (typed or pasted), rendered above the label.
- **Colour** — background hex.
- **Action** — one of:

Action type	What it does
Engine	<code>go / stop-all / panic / fade-out / goto (with cueIndex)</code>
Integration	Fire any cue command on any integration with explicit params
Panic	<code>stop-audio / clear-media / clear-lyrics / clear-tally / reset-timers / all-off</code>
Lock	Toggle / lock / unlock the show — disables every Stream Deck key except the unlock
Page	Switch the Stream Deck to another page
Live	Display-only — bind to current cue, lock state, or a timer ID

8.3 Pages

Configure multiple pages and add a **Page** action button to switch between them at runtime — useful when 32 keys still aren't enough.

8.4 Troubleshooting

- **No Stream Deck detected** — replug, check `system_profiler SPUSBDataType | grep -i elgato`. Some USB-C hubs drop HID; try a direct port.
- **Buttons paint but presses don't fire** — confirm the dispatcher is up by triggering a cue from the operator UI; the Stream Deck reuses the same dispatcher.
- **Buttons go blank after sleep** — WESC reconnects automatically every 3 s on disconnect events. If it doesn't recover, hit **Reconnect** in the integration card.

9. MIDI triggers (incoming)

WESC listens for incoming MIDI Note On (and Control Change) messages and maps each one to a cue index. Implementation: `easymidi` on top of the standard `CoreMIDI` framework.

9.1 IAC Driver setup for ProPresenter handoff

The most common pattern is to have ProPresenter fire MIDI on a slide change and have WESC pick it up. macOS uses an **IAC Driver** (Inter-Application Communication) as a virtual MIDI bus.

- Open **Audio MIDI Setup** (in `/Applications/Utilities`).

- **Window** → **Show MIDI Studio** (Cmd+2).
- Double-click the **IAC Driver** icon. If it's greyed out, tick **Device is online**.
- Add at least one Bus (default name "Bus 1"). Apply.
- In ProPresenter: **Preferences** → **Communication** → **MIDI**, pick the IAC Driver Bus, set MIDI Out On.
- In WESC: **Settings** → **Triggers** → **MIDI**, pick the IAC Driver bus from the input dropdown.

9.2 Configuration

Settings → Triggers → MIDI

Field	Default	Notes
Enabled	on	
Input	first available	Pick from the auto-detected list
Channel	all	0-15 (display 1-16) or null for all
Note offset	1	Note 1 → Cue 1; set 0 if your sender is 0-based

9.3 Mapping

- Note On (velocity > 0) → fire cue at index note - noteOffset.
- Control Change (value > 0) → same mapping using the controller number.
- Note On with velocity 0 (some senders use this in place of Note Off) is ignored.

9.4 Diagnostics

The integration card shows the **last received note** with its channel and velocity, refreshed live. Tap a key on a connected MIDI controller and confirm the readout moves before suspecting cue logic.

9.5 Troubleshooting

- **No inputs listed** — IAC Driver isn't online in Audio MIDI Setup.
- **Notes show in WESC but cue doesn't fire** — the cue index is out of range, or the cue is disabled.
- **All notes fire wrong cues** — check noteOffset. ProPresenter sends 1-based by default; WESC defaults to 1-based handling.

10. OSC triggers (incoming)

WESC listens for OSC messages over UDP and fires engine / cue actions in response.

10.1 Configuration

Settings → Triggers → OSC

Field	Default
Enabled	on
Port	53000
Bind address	inherits Settings → Network

10.2 Address pattern

Address	Args	Action
/wesc/go	—	GO (advance to next cue)
/wesc/goto	int index (0-based)	Jump to a cue by index
/wesc/cue/N	—	Fire cue N (1-based, taken from the URL)

Address	Args	Action
/wesc/stop	—	Stop engine
/wesc/reset	—	Full reset (engine + audio + media + tally)

10.3 Companion example

A typical Bitfocus Companion button to fire WESC cue 3:

- Action category: **OSC**
- Target: 127.0.0.1:53000 (or the WESC IP)
- Path: /wesc/cue/3
- Args: (none)

10.4 QLab example

In QLab, a Network cue with type **OSC Message**:

- Network destination: a UDP destination pointed at the WESC IP, port 53000.
- OSC Message: /wesc/go with no parameters.

10.5 Diagnostics

The integration card shows the **last received OSC message** including address, args, and source IP:port. Send a test from any OSC tool — if nothing shows, the packet isn't reaching the listening port.

10.6 Troubleshooting

- **Nothing arrives** — most often Settings → Network → Bind to is 127.0.0.1 and the sender is on a different machine. Switch to a LAN IP, or to "All interfaces", and confirm the firewall is letting UDP 53000 through.
- **Address rejected** — WESC's parser is strict about OSC type tags. If you're using a non-standard sender, test with QLab first to confirm the wire format.

11. HTTP trigger server (incoming)

WESC hosts an HTTP server (default port 7700) that exposes the engine over a small REST surface. The server also serves the embedded **Remote** UI and the **Stage** display at / and /stage respectively, so a phone or tablet on the show LAN can be a backup operator surface.

11.1 Configuration

Settings → Triggers → HTTP

Field	Default	Notes
Enabled	on	
Port	7700	
LAN mode	off	When off, listens on loopback only. When on, listens on the bind address from Settings → Network.
Auth token	—	Required when LAN mode is on; ignored on loopback. Send via Authorization: Bearer <token>, header X-WESC-Token: <token>, or ?token=<token>.

11.2 Endpoint reference

Method	Path	Action
GET	/	Remote UI HTML

Method	Path	Action
GET	/stage	Stage display HTML
GET	/api/status	Engine state (running, current cue, total cues)
GET	/api/cues	List of all cues with index, id, name, trigger, status
GET	/api/go	GO
GET	/api/goto/:index	Jump to cue by 0-based index
GET	/api/goto/name/:search	Jump to first cue whose name contains search (case-insensitive)
GET	/api/stop	Stop engine
GET	/api/reset	Full reset (engine + audio + media + tally)
GET	/api/stop-media	Clear media window only
GET	/api/stop-audio	Stop all audio only
GET	/api/license	Current licence state
WS	/ws	WebSocket: streams full state on connect, accepts {"type": "get-state"}

11.3 curl examples

```
# Loopback (no auth needed):
curl http://127.0.0.1:7700/api/status
curl http://127.0.0.1:7700/api/go
curl 'http://127.0.0.1:7700/api/goto/name/walk-on'
```

```
# LAN mode with token:
curl -H 'Authorization: Bearer abc123' \
  http://10.0.0.50:7700/api/reset
```

11.4 Panic from a hardware button

A hard-wired physical kill switch can be wired through any HTTP-capable controller (BrightSign, ShowControl, even an ESP32) — point it at GET /api/reset for a full kill, or /api/stop-audio for an audio-only cut.

11.5 Troubleshooting

- HTTP port 7700 already in use — another WESC instance, or another tool, owns the port. Change it in Settings → Triggers → HTTP.
- **401 Unauthorized** — LAN mode is on and no auth token configured / supplied. Either set a token in Settings or call from loopback.
- **CORS rejected from a browser app** — the server only mirrors Access-Control-Allow-Origin for same-origin requests (the served Remote/Stage pages). Cross-origin browser apps are intentionally blocked; use the WS endpoint or a server-side proxy.

12. NDI Output

WESC includes a **built-in NDI sender** that broadcasts the media output (videos, images, lyrics, slideshows, camera feeds) plus mixed audio as a single discoverable NDI source on the network. No third-party tools, NDI Tools, or virtual cable software are required. Implementation: the grandiose Node binding wrapping NDI SDK 5.

12.1 Configuration

Settings → NDI Output

Field	Default	Notes
Enabled	off	
Source name	WESC Media Output	What receivers see in their source list
Frame rate	50	25 / 30 / 50 / 60 fps

Field	Default	Notes
Include audio	on	If off, NDI is video-only
Auto-start on boot	off	Start the sender as soon as WESC launches
Groups	—	Comma-separated NDI Groups; empty for the public group

Manual controls: ■ **Start**, ■ **Stop**, **Test pattern**, **Bars ON / Bars OFF**.

Live status: state, frames sent, frames dropped, audio blocks sent, receiver count.

The global integration reconnect loop does not start NDI just because the integration is enabled. It only auto-starts the NDI sender when **Auto-start on boot** is on, so manual/per-cue NDI workflows stay quiet until a cue starts them.

12.2 Cue commands

Command	Params	Notes
ndiStart	sourceName?, fps?	Start broadcasting (or just keep going if already on)
ndiStop	—	Stop broadcasting
ndiTestPattern	duration? (s, default 5)	Fire-and-forget timed SMPTE bars + tone
ndiTestPatternOn	—	Indefinite SMPTE bars + 1 kHz tone
ndiTestPatternOff	—	Stop the indefinite test pattern

The persistent **Bars ON / Bars OFF** pair is the production pattern: pre-roll a "Bars ON" cue, the engineer downstream confirms the line-up, the next cue fires "Bars OFF" and the show starts.

12.3 Audio mixer architecture

WESC sums up to three concurrent PCM sources into the NDI audio stream via `ndiMixer`:

- **Audio cues** — the renderer's `<audio>` elements are tapped via Web Audio (`AudioContext` → `AudioWorkletNode`) and forwarded to main as planar Float32 stereo blocks.
- **Video file audio** — when `playVideo` fires, an `ffmpeg` subprocess (paced with `-re`) decodes the audio track in parallel with renderer playback and feeds PCM blocks into the mixer.
- **Camera audio** — `showCamera` cues with `audio: true` capture the `getUserMedia` audio track via Web Audio in the media renderer and feed the mixer.

All three are sum-and-clipped into a single planar Float32 block per send, with a per-source ring buffer (max 4 blocks ≈ 85 ms latency cap, drop-oldest overflow) so a stalled source can't back up the others.

Audio Player cues have a cue-level `audioOutput` parameter:

- `default` (or blank) — the audio cue plays through the selected/default local audio output and WESC also mirrors it into NDI Output when the sender is running.
- `ndi` — WESC connects the audio cue to the NDI tap only and does not connect it to the local speaker/device destination. If the NDI tap cannot initialise, the cue does not fall back to local playback.

Audio play supports `trimStart` and `trimEnd` in seconds. `Audio playPlaylist` accepts either plain file paths or item objects:

```
{ "file": "/path/to/song.mp3", "volume": 80, "trimStart": 12.5, "trimEnd": 94, "fadeIn": 0.5, "fadeOut": 2 }
```

For playlist items, `trimStart`, `trimEnd`, `fadeIn` and `fadeOut` are per-track. Blank or missing `trimStart` means start at the beginning; blank or missing `trimEnd` means play to the end. Blank fades mean no per-track fade. A top-level `fadeIn` on `playPlaylist` still applies to the first track when that first track has no per-track fade in.

Audio file paths are resolved defensively at playback time. If a stored filename has drifted to include a leading track number or cue UUID, WESC will look for the matching original filename in the same folder before failing the cue.

Video cues also have a cue-level `audioOutput` parameter:

- `default` (or blank) — the media renderer plays local audio and WESC also mirrors the decoded video audio into NDI Output when the sender is running.
- `ndi` — the media renderer mutes the local `<video>` element, but the `ffmpeg` decode path still feeds the NDI mixer. Use this for video playback that should be heard only by downstream NDI receivers.

`muted=true` is different: it disables video audio everywhere, including the NDI audio path.

12.4 Bandwidth

Video is BGRA at sender frame rate (50 / 60 fps for HD is roughly 100-200 Mbit/s on the wire — NDI compression). Audio is planar Float32 stereo at 48 kHz. A single NDI sender on a gigabit show LAN is unproblematic; multiple senders on the same LAN at high frame rate / high resolution can saturate switches without adequate backplane capacity. NDI Discovery Server on a dedicated machine is recommended on busy networks.

12.5 Receivers and groups

The integration card shows live receiver count via `sender.connections(0)`. Use NDI **Groups** to scope visibility — receivers on a different group won't see the source. Empty groups field means "public" (visible to everyone).

12.6 Mixing partner integrations

WESC's NDI output can be received by any NDI-aware tool — OBS (with the OBS-NDI plugin), vMix, Wirecast, Resolume, TriCaster — as a regular NDI source. No additional WESC config beyond enabling the sender.

For the reverse direction (OBS / vMix output flowing into WESC), use the NDI Input integration (next chapter).

12.7 Troubleshooting

- `grandiose native module failed to load` — the build of `grandiose` for your Electron version is broken. `npm install && npx electron-rebuild` from the `wesc` folder.
- **No receivers see the source** — check **Groups** (mismatched group is the most common cause), and that mDNS / Bonjour traversal works between the WESC machine and the receiver.
- **Audio meter flat at receiver** — almost always a timecode bug in the audio frames. WESC explicitly omits the `timecode` field on audio frames; if you've patched the integration locally make sure you haven't reintroduced `timecode: 0n` (which makes receivers ignore every packet as a duplicate).
- **Audio bursting at receiver** — `ffmpeg` pacing. Verify the subprocess is started with `-re` before `-i`; without it `ffmpeg` dumps tens of seconds of PCM in milliseconds.

13. NDI Input (NEW in v1.7)

WESC v1.7 receives remote NDI sources as first-class cue sources. Translator microphones, remote cameras, instrument feeds, interview line feeds — anything broadcasting NDI on the network — can flow into WESC and be routed through the same fader / meter pipeline as file cues.

13.1 What's required at the sender end

Any NDI 5+ sender on the same LAN: NDI Tools test pattern, OBS with the OBS-NDI plugin, vMix output, hardware NDI cameras, iPhone NDI Camera apps, another WESC instance.

13.2 Settings → Audio → NDI Input

Field	Notes
Source picker	Click Refresh to scan the network; pick a source
Output	Same dropdown as audio cues — pick the device + channels
Connect / Disconnect	Live ad-hoc bring-up outside of cue control
Extra IPs	Comma-separated IPs of senders mDNS can't reach
Groups	Comma-separated NDI group names; blank = public

13.3 Network requirements

NDI uses **mDNS / Bonjour** by default. Senders and receivers on the same Layer-2 segment usually find each other automatically. mDNS does not cross routers / VLANs.

When the sender is across a VLAN boundary, on isolated Wi-Fi, or behind AP isolation, use one of:

- **Extra IPs** — manually add the sender's IP to WESC's **Settings** → **Audio** → **NDI Input** → **Extra IPs** field. WESC then connects directly to the sender, bypassing mDNS discovery.
- **NDI Discovery Server** — run NewTek's Discovery Server on a third machine (free with NDI Tools). Both sender and receiver register with it via the `NDI_DISCOVERY_SERVER` environment variable, removing the mDNS requirement entirely.

13.4 Cue commands

Command	Params	Notes
<code>ndiInput.start</code>	<code>sourceName, sourceUrlAddress, outputDevice?, outputChannels?, videoOutput? (none / media), fit? (contain / cover), instanceId?</code>	Open a receiver
<code>ndiInput.stop</code>	<code>instanceId? (or all / blank for stop-everything)</code>	Close one or all receivers

The cue editor's **Source** picker uses live discovery; the source's `urlAddress` is what's actually persisted. `name` is for display.

13.5 Multi-instance

Run multiple NDI inputs concurrently — translator microphone + remote camera + interview feed — each on its own audio bus on the Mixer page. Each instance is keyed by `instanceId`:

- Set the cue's optional **Instance name** to a stable identifier (e.g. `translator-en, interview-line`).
- If left blank, WESC sanitises the source name into an instance ID.
- Two cues with the same `instanceId` replace one another (the second `start` stops the first).

The practical concurrent-receiver limit on a typical Mac is 3-4 instances. Each receiver is its own grandiose `Receiver` object with its own frame loop.

13.6 Video output

Set **Video Output** to `media` to paint the source's video on the media canvas. Only one receiver at a time can own the canvas — the most recent `start` cue with `videoOutput=media` claims ownership. Other receivers continue to send audio but their video frames are discarded.

Pixel format is **RGBA** (`COLORFORMAT_RGBX_RGBA`), so `putImageData` blits straight to the canvas with no per-pixel byte-swap.

13.7 Bandwidth modes

- Cues with `videoOutput=none` open the receiver in NDI **AUDIO_ONLY** bandwidth mode — cheap on the network, useful for translator feeds.
- Cues with `videoOutput=media` open the receiver in **HIGHEST** bandwidth mode for full-quality video.

13.8 Per-instance audio router stream IDs

Each NDI input instance gets a stream ID of the form `ndi-input:<instanceId>` in the audio router. Bus assignment on the Mixer page tracks per-instance, so the translator and the interview feed have independent faders even when both are technically NDI input cues.

13.9 Stop-all behaviour

The **Stop All Audio** panic action and the Live page **STOP ALL** button stop every running NDI input instance and clear the media canvas if NDI was painting it. This is intentional — a panic should kill remote feeds the same way it kills local audio.

13.10 Troubleshooting

- **No sources appear on Refresh** — mDNS isn't reaching the sender. Add the sender's IP to **Extra IPs**.
- **Source listed but won't connect** — the sender's `urlAddress` is reachable from WESC but the actual NDI port (5961+) is firewalled. Open NDI ports on the path.
- **Audio works, video black** — the cue's `videoOutput` is none, or another instance has claimed canvas ownership.
- **Non-audio data received on audio capture** — pre-1.7 builds had separate `audio()` / `video()` loops. The current implementation uses unified `receiver.data()` with type dispatch; if you see this error, you're on an old build.

14. Audio routing & multi-channel devices (NEW in v1.7)

Per-cue audio routing into multi-channel devices is a v1.7 headline feature. The architecture is built around a single owner — `audioRouter` — that all audio sources feed into.

14.1 Setup recipes

Dante Virtual Soundcard

The most common production setup. Install Dante Virtual Soundcard, license for 16 / 32 / 64 channels as needed, configure routes in Dante Controller. In WESC: **Settings** → **Audio** → **Default device** → **Dante Virtual Soundcard**. WESC sees as many channels as DVS exposes to the OS.

Loopback (Rogue Amoeba)

Build a virtual aggregate device combining multiple physical outputs into one channel-rich device, then pick it as WESC's default. Useful when the show needs FOH, backstage, and translation routed to three different physical interfaces.

macOS Aggregate Devices

Audio MIDI Setup → **Aggregate Device** combines built-in output, USB DACs, Thunderbolt audio interfaces into a single multi-channel device WESC can address as one output. Free, no extra licences, but less flexible than DVS or Loopback.

14.2 Channel naming

Settings → **Audio** → **Channel names** lets you assign a friendly alias to each channel pair (and each single channel for mono routes):

- Ch 1-2 → FOH
- Ch 3-4 → Translation EN
- Ch 5 → Talkback (mono)
- Ch 6 → Stage Monitor (mono)
- Ch 7-8 → Recording

Aliases appear in every channel-pickable UI (per-cue Output dropdown, Mixer strips, NDI Input output dropdown). They live with the runsheet so the names stick.

14.3 Per-cue Output dropdown

Every audio cue gains an **Output** dropdown that overrides the default for that one cue:

- **Use default (Settings → Audio)** — the global default.

- **Ch N-N+1** — <alias> — a stereo pair.
- **Ch N** — <alias> (**mono**) — a single channel; WESC sums L+R to mono.

Channels shown depend on the current default device's channel count.

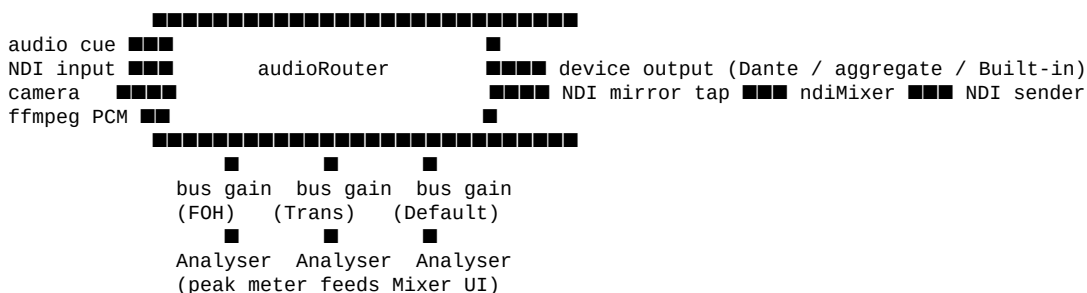
14.4 The Mixer page

A top-level **Mixer** tab (between Timers and Settings) is a runtime mixing console. One vertical fader strip per named bus, plus a master "Defaults" strip catching everything routed to "Use default":

- **Faders** ride the level on every active cue currently routed to the bus, in real time. 30 ms ramp prevents zipper noise. The fader sits *post* each cue's own volume — it's a master per bus.
- **Mute** hard-cuts the bus.
- **Peak meters** with peak-hold dot, green up to ~-15 dB, amber from -10, red above 0. The white peak-hold dot lingers ~1.5 s after the last peak then decays.
- **dB scale** beside each fader: $-\infty$, -20, -10, 0, +6.

Faders persist per output device. Switching the default device in Settings recalls a different saved fader set for that device.

14.5 Routing pipeline architecture



Concrete components inside `src/renderer/modules/audioRouter.js`:

- A per-device **Web Audio context** so each output device runs at its native sample rate.
- A **ChannelMergerNode** that maps virtual channel pairs onto physical device channels.
- One **bus gain node** per named bus, controlled by Mixer faders.
- One **AnalyserNode** per bus, sampled by the Mixer UI for live peak metering.
- The **NDI mirror tap** pulls a parallel stereo sum of all sources for the NDI sender's audio path.

Routing limitations:

- WESC writes channel data via Web Audio's `ChannelMergerNode`. Devices reporting fewer than 8 channels work without configuration; high-channel-count devices may need the device's own routing software (Dante Controller, Loopback) configured to expose the desired channel count to the OS first.
- Aggregate devices on macOS sometimes report fewer channels than expected if the constituent devices are at different sample rates. Force a common rate in Audio MIDI Setup.

15. AI lyric tools (NEW in v1.7)

The Song Editor gains optional AI-assisted lyric clean-up and stanza formatting, powered by Claude.

15.1 Setup

- Sign up at [\[console.anthropic.com\]](https://console.anthropic.com)(<https://console.anthropic.com>).
- Create an API key.

- Paste it into **Settings** → **AI** → **Claude API key**.
- Optionally pick a model — Claude Sonnet 4.6 (recommended), Haiku 4.5 (cheaper / faster), Opus 4.6 (most thorough).
- Hit **Test connection**.

15.2 What it powers

Two new buttons appear next to the Lyrics label in the Song Editor when an API key is configured:

- **Tidy** — strips timestamps, normalises curly quotes, fixes ALL CAPS lines, removes section labels like [Verse 1].
- **Auto-stanza** — splits a wall of lyrics into verse / chorus / bridge stanzas separated by blank lines.

Both buttons show a side-by-side diff preview before applying.

15.3 Cost

Lyrics are short — a typical song clean-up costs fractions of a penny. A full Sunday-morning service of 6 songs is well under £0.05.

15.4 Privacy

- **Lyrics are sent to Anthropic only when you press Tidy or Auto-stanza.** No background telemetry, no automatic processing.
- **The API key is stored locally** in the WESC config file. It is never transmitted to Whiteley Events.
- **Whiteley Events receives nothing** from this feature — no metadata, no lyric content, no usage statistics.

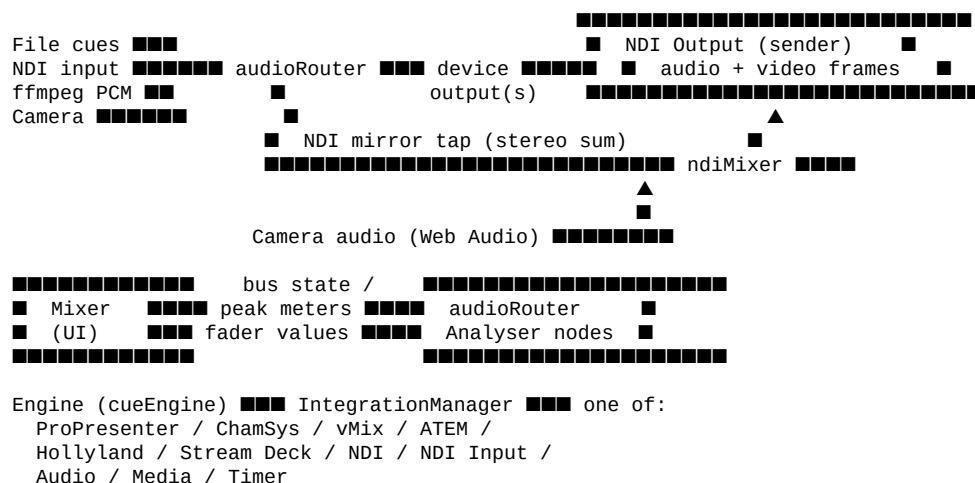
This is also documented in section 3.1 of the Privacy Notice (bumped to v1.1.0 to cover this feature).

15.5 CCLI / copyright caveat

For worship use: the AI tools manipulate text you've already entered into WESC. Whether the source lyrics are yours to use is a separate question governed by your CCLI / OneLicense / equivalent licence. Sending lyrics to Anthropic for re-formatting is unlikely to constitute a public performance, but the canonical answer depends on your jurisdiction's copyright law and your licence terms — ask your licensing body if you're unsure.

16. Architecture overview

16.1 Signal flow (text diagram)



16.2 Single ownership

The architectural rule for v1.7 is **audioRouter owns audio output**. Anything emitting PCM — file cues, NDI input audio, ffmpeg-decoded video audio, camera audio — feeds into the same router. There is no longer a separate `_ndiCueCtx` or ad-hoc `<audio>` plumbing. This guarantees a single point of fader control, a single peak-meter feed for the Mixer, and a consistent NDI mirror tap regardless of source.

16.3 IPC boundary

- Audio decoding for NDI uses an **ffmpeg subprocess** spawned from main, paced with `-re`, output as planar Float32 1024-sample blocks. Not part of the renderer at all.
- NDI input frames cross IPC main → renderer as `{event: 'audio' | 'video', instanceId, ...}` envelopes.
- Cue command results round-trip main → integration → main → renderer.

17. Diagnostics

17.1 Connection Status

Settings → **General** → **Connection Status** is a one-pane status board for every integration, showing:

- Enabled / disabled toggle.
- Live connection state.
- Last error string, if any.
- Per-integration receiver / client / subscriber counts where applicable.

When WESC starts, it runs one startup **Test** pass across enabled external integrations so stale red states can recover without the operator opening Settings. Enabled external integrations are also retried automatically while they remain disconnected. ProPresenter, ChamSys and vMix use the same connection probe as their **Test** button; Hollyland and NDI use their start routines; ATEM and Stream Deck rebuild their connection handles. Internal integrations and NDI Input are excluded because their red/green state depends on active cue/runtime state rather than a single always-on device connection.

17.2 Show Readiness

Loading a runsheet runs a **Show Readiness** pass that warns or fails when:

- Required media/audio files are missing or unsupported.
- Timed cues include seconds instead of the supported HH:MM format.
- Duplicate MIDI Cue IDs are present.
- WESC Control `goTo` or Operator `showCue` targets cannot be resolved.
- ProPresenter `triggerPlaylistItemByName` is blank.
- A runsheet has NDI cues but the NDI Output integration is disabled.
- The grandiose native module failed to load.
- NDI is enabled but not broadcasting and no `ndiStart` cue exists.
- NDI is broadcasting with zero subscribed receivers.
- NDI-only audio/video output is requested while NDI Output is disabled or unavailable.
- Audio playlists are empty or contain invalid trim/fade values.
- WESC Control `goTo`, `Follow` or `Auto-continue` links can form a cue loop.
- An enabled external integration has not yet reported a successful startup/reconnect watchdog check.

The top bar **Readiness** pill opens the panel. The panel can also export a diagnostic JSON bundle containing the current app version, runsheet path, readiness results, integration states, watchdog timestamps, recent command

history, trigger status and recent show log events.

The panel includes an **Audio cue audit** table so operators can see which cues can make sound, whether they are routed locally or NDI only, and whether they are muted, looped, shuffled, trimmed or faded.

17.3 Logs

Console output goes to:

- The Electron renderer DevTools console (View → Toggle Developer Tools).
- The main process console — visible via the macOS **Console** app filtered to WESC, or by launching `WESC.app/Contents/MacOS/WESC` from a Terminal.
- Persistent show logs at `~/Library/Logs/WESC/` (rotated per-show).

Every integration emits prefixed log lines ([WESC ATEM], [WESC NDI input], [WESC ChamSys], etc.) for grep-friendly post-show analysis.

18. Troubleshooting

NDI source not appearing

- Confirm sender and WESC are on the same Layer-2 network (or use Extra IPs / Discovery Server).
- Check **Groups** match between sender and receiver — non-public groups are scope-limited.
- Disable AP isolation on Wi-Fi.
- Add the sender's IP to **Settings** → **Audio** → **NDI Input** → **Extra IPs** if mDNS can't traverse.

ATEM not connecting

- Confirm port 9910 is open and reachable (not 9911 — that's the simulator).
- Confirm only one ATEM control client at a time per switcher (some firmwares enforce this).
- Try simulator mode (mode: `simulator`, port 9911) to isolate WESC from network issues.

Stream Deck not detected

- Replug. Try a direct USB port (not a hub).
- Confirm the device shows in `system_profiler SPUSBDataType | grep -i elgato`.
- If the library failed to compile, `npm install && npx electron-rebuild` from the wesc folder.
- Quit any other Stream Deck-aware app (Elgato desktop, Companion) — only one process can hold the HID handle.

MIDI not receiving

- Audio MIDI Setup → MIDI Studio → IAC Driver → Device is online ticked.
- IAC Bus added with at least one bus.
- Sender (ProPresenter, Companion, etc.) configured to send to that exact bus.
- WESC's MIDI input dropdown showing the bus, not "(none)".

OSC silence

- Settings → Network → Bind to is set correctly (usually a LAN IP, not loopback).
- Firewall is allowing UDP on the OSC port.
- The sender is hitting the right port (53000 by default).

- The integration card's **Last received message** confirms packets are arriving — if it stays empty, the issue is below WESC.

HTTP triggers 401

- LAN mode is on but no auth token configured. Either disable LAN mode (loopback only, no auth needed) or set a token in Settings → Triggers → HTTP and pass it via Authorization: Bearer <token>.

Audio dropping out under load

- Open Activity Monitor — if WESC's CPU is pegged, lower the NDI sender frame rate or reduce concurrent NDI input instances.
- For Dante Virtual Soundcard, confirm latency is set high enough (typically 10 ms is fine; drop to 4 ms only on a quiet show LAN).

Hollyland HUB won't connect

- HUB switcher mode set to vMix.
- HUB's switcher IP matches the WESC machine's show-LAN IP.
- HUB's switcher port matches WESC's Hollyland port (8099).
- Hollyland integration enabled in WESC.
- Firewall allowing inbound TCP 8099.

Appendix A — Full cue command reference

Integration	Command	Params
propresenter	triggerNext	—
propresenter	triggerPrevious	—
propresenter	triggerSlide	playlistIndex, presentationIndex, slideIndex
propresenter	triggerPlaylistItemByName	playlist, item
propresenter	triggerFocusedPlaylistItem	index
propresenter	triggerPlaylistItem	playlistId, index
propresenter	triggerFocusedPresentationGroup / triggerActivePresentationGroup	groupId
propresenter	triggerPresentationGroup	presentationId, groupId
propresenter	clearAll	—
propresenter	clearSlide	—
propresenter	triggerPlaylist	id
propresenter	triggerMedia	id
propresenter	startTimer / stopTimer / resetTimer	id
chamsys	fireCue	playback, cue
chamsys	goPlayback / stopPlayback / releasePlayback / activatePlayback	playback
chamsys	setLevel	playback, level (0-100)
chamsys	setIntensity	channel, level (0-100)
vmix	cut	—
vmix	fade	duration (ms)
vmix	transitionToInput	input, mix
vmix	setInputPreview	input

Integration	Command	Params
vmix	over layOn	over lay (1-4), input
vmix	over layOff	over lay
vmix	startRecording / stopRecording / startStreaming / stopStreaming	—
vmix	playInput / pauseInput	input
vmix	setVolume	input, volume (0-100)
atem	connect / disconnect / syncHollyland	—
hollyland	setProgram / setPreview / setOff	channel (1-16)
hollyland	clearAll	—
hollyland	set	channel, state
streamdeck	switchPage	page
streamdeck	flashButton	index
streamdeck	reconnect	—
audio	play	file, volume, fadeIn, trimStart, trimEnd, loop, audioOutput? (default / ndi), outputDevice?, outputChannels?
audio	playPlaylist	files (paths or { file, volume, trimStart, trimEnd, fadeIn, fadeOut } items), volume, loop, shuffle, audioOutput? (default / ndi), outputDevice?, outputChannels?
audio	stop / pause / resume / next	—
audio	fadeOut / fadeNext	duration
audio	setVolume	volume
media	showImage	file, duration, fit
media	playVideo	file, volume, audioOutput? (default / ndi), loop, fit, muted?
media	showSlideshow	playlist, duration, loop
media	playVideoPlaylist	playlist, volume, audioOutput? (default / ndi), loop, fit, muted?
media	showLyrics	text, bgColour, bgImage, fontSize
media	showCamera	deviceId, label, audio, fit
media	clear	—
ndi	ndiStart	sourceName?, fps?
ndi	ndiStop	—
ndi	ndiTestPattern	duration? (s)
ndi	ndiTestPatternOn / ndiTestPatternOff	—
ndiInput	start	sourceName, sourceUrlAddress, outputDevice?, outputChannels?, videoOutput?, fit?, instanceId?
ndiInput	stop	instanceId?
operator	showMessage	title?, text, tone? (info / warning / urgent)
operator	showCue	targetCueId (preferred) or target, text?
operator	showImage	file, title?, text?, fit? (contain / cover)
operator	clear	—
operator	speakViewOnly	text, voice?, destination?, outputDevice?, outputChannels?

Integration	Command	Params
timer	start / pause / reset / stop	timerId
timer	stopAll	—

Appendix B — Runsheet JSON schema for cue actions

A WESC runsheet is JSON. Each cue has an actions array, where every entry is an action object of the shape:

```
{
  "integration": "<integration name>",
  "command": "<command id>",
  "params": { /* command-specific params */ }
}
```

Worked examples:

Fire a ProPresenter slide on a cue

```
{
  "integration": "propresenter",
  "command": "triggerSlide",
  "params": {
    "playlistIndex": 0,
    "presentationIndex": 2,
    "slideIndex": 5
  }
}
```

Walk-on music to the FOH bus

```
{
  "integration": "audio",
  "command": "play",
  "params": {
    "file": "/Users/operator/show/walk-on.wav",
    "volume": 80,
    "loop": false,
    "outputDevice": "DanteVirtualSoundcard",
    "outputChannels": "1-2"
  }
}
```

Translation feed from a remote NDI source

```
{
  "integration": "ndiInput",
  "command": "start",
  "params": {
    "sourceName": "TRANSLATOR-MAC (TranslatorBooth)",
    "sourceUrlAddress": "10.0.0.42:5961",
    "outputDevice": "DanteVirtualSoundcard",
    "outputChannels": "3-4",
    "videoOutput": "none",
    "instanceId": "translator-en"
  }
}
```

Send a stage-manager message to the Operator screen

Open the Operator screen from Remote Access or directly at <http://<wesc-ip>:3030/operator>, then fire:

```
{
  "integration": "operator",
  "command": "showMessage",
  "params": {
    "title": "Standby",
    "text": "Choir to stage left in 2 minutes.",
    "tone": "warning"
  }
}
```

Send AI operator voice to Dante/audio channels

The same Operator voice command can target the host Mac's local speaker or a production audio output route instead of View Only browsers:

```
{
  "integration": "operator",
  "command": "speakViewOnly",
  "params": {
    "text": "Band, please return to stage.",
    "voice": "alloy",
    "destination": "localSpeaker"
  }
}
```

For Dante/channel routing:

```
{
  "integration": "operator",
  "command": "speakViewOnly",
  "params": {
    "text": "Stage manager, stand by for band entrance.",
    "voice": "alloy",
    "destination": "audioOutput",
    "outputDevice": "DanteVirtualSoundcard",
    "outputChannels": "5"
  }
}
```

In the cue editor, the same action exposes English-labelled voice presets and a **Test Voice** button. The test path uses the configured destination, so View Only tests require a connected View Only browser, local-speaker tests play on the host Mac, and audio-output tests require the selected non-default device/channel route.

For safety, WESC rejects Dante/channel mode unless the effective route resolves to a non-default output device and a specific channel or channel pair. It will not fall back to the app host computer's default audio output. Use `destination: "localSpeaker"` when local host playback is intentional.

For cue cards, use `operator.showCue` with `targetCueId` so the target survives cue reordering.

Cross-fade to ATEM camera 2 with auto-tally

(Assumes ATEM Auto-Hollyland is enabled and mappings have ATEM 4 → Tally 2.)

```
{
  "integration": "vmix",
  "command": "fade",
  "params": { "duration": 1000 }
}
```

The ATEM and Hollyland integrations are passive — they follow the switcher state automatically rather than being driven by cue actions, so the cue itself stays focused on the visible transition.

Pre-show NDI bars

```
[
  {
    "integration": "ndi",
    "command": "ndiStart",
    "params": {}
  },
  {
    "integration": "ndi",
    "command": "ndiTestPatternOn",
    "params": {}
  }
]
```

Followed in a later cue by:

```
{
  "integration": "ndi",
  "command": "ndiTestPatternOff",
  "params": {}
}
```

Fire cue 3 from Companion via OSC

Companion-side OSC button → `127.0.0.1:53000 → /wesc/cue/3` → no args. WESC fires the cue at index 2 (the third cue in the runsheet).

Panic-stop everything from a hardware button

GET `http://10.0.0.50:7700/api/reset` → engine reset, audio stopped, media cleared, tally cleared.

End of WESC Integration Guide v1.7.212.