USER MANUAL

ACE World.

A 3D virtual venue where your audience joins as avatars, watches your live feed together, and hears each other in spatial voice.

Gather every room.

Web · Desktop · VR

6 sections

ace-presenter.app

Contents

OVERVIEW

ACE Virtual World — Preview Guide

A shared 3D virtual venue that replicates a live event. Not a video on a screen — a space your audience enters: they watch your live stage feed togeth…

ACE Virtual World — Preview Guide

A shared **3D virtual venue** that replicates a live event. Not a video on a screen — a space your audience enters: they watch your live stage feed together, see one another as avatars, hear each other in **spatial voice**, and a **producer** drives the room in real time. It runs in a desktop app, in any modern browser, and inside a Meta Quest headset via WebXR.

> **In development — not yet released.** ACE Virtual World is being built now. This is a **preview guide**: it describes the intended experience and is honest about what already works in early builds versus what is still being built. Nothing here is a promise of a ship date, and details will change before release. To be brought in as it opens up, register interest.

How to read this guide

This guide has **two readers** in mind, and every chapter says which parts are for whom:

- **The operator / producer** — the licensed person who composes the venue and runs the show. Setting up a venue, feeding the stage, and driving the room are operator tasks.
- **The attendee** — anyone you invite. They join by opening a link, with **no account, no app install, and no license**. If you were sent a link, Joining as an Attendee is the only chapter you need.

Status vocabulary. Because this is a pre-release product, features are flagged inline so you always know what to expect today:

BADGE	MEANING
`preview · verified`	Working in early builds and confirmed end-to-end on real infrastructure
`preview`	Built and running in early builds; still being hardened, and not yet perceptually or interactively signed off
`in development`	Actively being built; the plumbing exists but the finished experience is not here yet
`planned`	Designed and on the roadmap; not started or early scaffold only

When a feature's badge and the surrounding text disagree, trust the text — it carries the detail.

The mental model: an event is five layers

Everything in ACE Virtual World maps onto one of five layers. Keeping them straight makes the rest of the guide click into place — and each layer is at a different stage of maturity.

#	LAYER	WHAT IT IS	WHERE IT IS TODAY
1	**Ingest**	Where your content comes from — OBS, vMix, a camera, a webcam, ACE Presenter	Live camera and switcher feeds work `preview · verified`; ACE Presenter integration is `planned`
2	**Transport**	Real-time delivery of video, voice, and spatial audio	One WebRTC stack (LiveKit) carries all three; live media is `preview · verified`, spatial voice is `preview`
3	**World**	The venue itself — stage, floor, screens, breakout areas	A composable graybox venue with multiple surfaces works; finished venue **art (GLTF)** is `in development`
4	**Presence**	The people — avatars, movement, spatial voice	Avatars, movement, and positioned voice run; final avatar **art** and lip-sync rigs are (in development)
5	**Production**	Show control — scenes, cues, lighting, audio, moderation	The producer console drives scenes, cues, lighting moods, audio, and zones `preview`

Read the layers top to bottom and you have the whole pipeline: a feed comes **in** (Ingest), travels over **one transport**, lands in a **venue** full of **people**, and a **producer** shapes the moment for everyone.

What already works, and what is still coming

A quick, honest snapshot. The chapters go deeper.

Working in early builds

- **Live media** — OBS / vMix / a hardware encoder → LiveKit → the stage screen, sub-second. Confirmed end-to-end on real LiveKit Cloud. `preview · verified`
- **The producer console** — scenes (Intro / Main stage / Breakout), a firing cue list, lighting moods, feed routing, and zone lock/unlock, all synced to every client. `preview`
- **Spatial voice** — each attendee's voice is positioned in 3D at their avatar and attenuates with distance. Built and running; needs a session with several mic'd people to sign off perceptually. `preview`
- **Audience join** — a link plus a short code, no account, in a browser or a Quest headset. `preview`

Still being built

- **Venue and avatar art** — the venue is a functional graybox and avatars fall back to simple capsules until finished 3D art (GLTF/GLB) is dropped in. `in development`
- **Temporal A/V sync** — frame-accurate alignment of the stage feed with cues across every attendee's screen. (in development)
- **Captions** — live captions of the stage audio via speech-to-text. A local-microphone demo exists; the production caption source is `in development`.
- **Production billing / licensing** — the metered "go live on your license" path is wired but rides on account infrastructure still being finished. `in development`

For the full, layer-by-layer status, see Platforms, Requirements & Status.

Table of contents

Where ACE Virtual World fits in the suite

ACE Virtual World is part of the **ACE Suite** — one account, one brand, one cueing language across the products. The plan is that you schedule an event in ACE Schedule, present with ACE Presenter, and gather the room in ACE Virtual World, all speaking the same feed and cue model. That cross-product handoff — ACE Presenter publishing straight into a world surface — is `planned`; see Feeding the Stage ▸ Where ACE Presenter fits.

SECTION 01

1. Getting Started & Concepts

This chapter explains what ACE Virtual World is, the ideas the rest of the guide builds on, and the shape of a typical session — so both operators and…

01 · 1. Getting Started & Concepts

This chapter explains what ACE Virtual World is, the ideas the rest of the guide builds on, and the shape of a typical session — so both operators and attendees know what they're stepping into.

> **Preview product.** ACE Virtual World is in development and not yet released. This chapter describes the intended experience; feature badges mark what already works in early builds.

What ACE Virtual World is

Picture a real event at a real venue — a stage, a big screen, a floor where people stand, a couple of breakout areas off to the side. ACE Virtual World builds a **virtual twin** of that kind of space and drops your **live event** inside it:

- The **stage screen** plays your live camera or switcher feed, sub-second, the same way it would on a video wall in the room.
- **Attendees** join as **avatars** and stand together in the space — they see each other, not names in a chat list.
- Everyone hears each other in **spatial voice**: a voice comes from the direction of the person speaking and gets quieter with distance, so the room sounds like a room.
- A **producer** drives the moment for everyone — changing scenes, firing cues, adjusting lighting and the audio mix, opening and closing areas.

It is deliberately **not** "a video call" and **not** "a video on a page." It is a shared place.

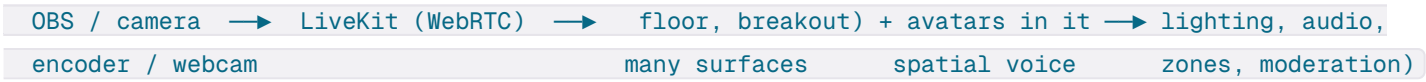
Two things it is not (yet)

Being honest up front: today the venue is a clean, functional **graybox** — real geometry with a stage, screens, a floor, and breakout zones, but not the finished, art-directed venue. Finished venue and avatar **art** is in development. And **frame-accurate** alignment of the feed with cues across every screen (temporal sync) is still being built. Everything else in this chapter describes what already runs.

The five layers (the whole system in one picture)

The README introduces the five-layer model; here it is as a flow you can hold in your head:

```
  1. INGEST            2. TRANSPORT          3. WORLD          4. PRESENCE        5. PRODUCTION
 where content     one real-time stack     the venue         the people          show control
 comes from        (video+voice+audio)     (stage, screens,  (avatars, voice)    (scenes, cues,
```



- **Ingest** is your job as operator: point a camera or switcher at the world. See Feeding the Stage.
- **Transport** is handled for you. A single WebRTC media server (LiveKit) carries the live video, everyone's voices, and the spatial-audio positioning — one stack instead of three.
- **World** is the venue you compose: surfaces (screens), lights, spawn points, breakout zones, branding. See Setting Up a Venue.
- **Presence** is the crowd — each attendee is an avatar that moves, and whose voice is positioned at their location.
- **Production** is the live show, run from the producer console. See Going Live & the Producer Console.

You don't have to master all five to run a simple event. The minimum is: compose or open a venue (World), point OBS at it (Ingest), and press **Go live** (Production). The rest deepens the experience.

Who does what: operator vs attendee

ACE Virtual World has two roles, and the difference is enforced, not cosmetic — an attendee never sees or can trigger operator controls.

The operator (a.k.a. producer)

The **licensed** person who sets up and runs the venue. As operator you:

1. **Compose** the venue — place screens, set branding, lay out scenes. (See Chapter 2.)
2. **Feed** the stage — connect OBS / vMix / a camera / a webcam. (See Chapter 3.)
3. **Go live and run the show** from the producer console — scenes, cues, lighting, audio mix, zones, moderation. (See Chapter 4.)

The operator tools live in a **composer** (design the room) and a **producer console** (run the room). On the native desktop app the producer console is its own always-on-top window.

The attendee

Anyone you invite. Attendees:

- **Open a link** (and sometimes type a short join code) — nothing to install, no account, no license.
- **Watch** the stage feed, **look around** the venue, **move**, and **talk** in spatial voice.
- Join from a **browser** (desktop or mobile) or step inside on a **Meta Quest** headset via WebXR.

Everything an attendee needs is in Chapter 5.

> **Licensing, briefly.** The **operator** is the licensed party. The **audience is free** — they never sign in. When an operator goes live, that session is intended to be metered against the operator's ACE license. That production billing path is wired but rides on account infrastructure still being finished — see Status. (in development)

The shape of a session

Here is a whole event end to end, so the pieces have a place to hang.

Before the event (operator, minutes to hours ahead)

1. Open or compose a venue: choose the space, place your stage screen(s) and any side screens, set your logo and colours, and lay out a few scenes (e.g. Intro, Main stage, Breakout). → Chapter 2
2. Decide how the stage feed gets in — usually **OBS or vMix → LiveKit**. Have that ready to start streaming. → Chapter 3

Going live (operator)

1. Press ● **Go live** on the operator console. This creates the event as its own isolated world and shows you a **join code** and a **copyable link**.
2. Share the link with your audience (email, chat, a page — however you reach them).
3. Start your OBS/vMix stream. Switch the stage feed to **live**; the stage screen lights up.

During the event

1. Attendees open the link and appear as avatars. They watch, move, and talk.
2. You drive the room from the producer console: change **scenes**, fire **cues**, set **lighting moods**, balance the **audio mix** (show audio vs. crowd voices), and open or close **zones**. Mute the crowd for a keynote, or mute one person for moderation. → Chapter 4

Ending

1. Press **End event** to stop new joins and close the world.

Multiple operators can run **concurrent** events — each event is a completely separate world with its own people, scene, feeds, and audio mix.

Where to go next

- **Operators**, start with Setting Up a Venue.
- **Attendees**, jump straight to Joining as an Attendee.
- Want the candid maturity picture first? See Platforms, Requirements & Status.

SECTION 02

2. Setting Up a Venue

For the operator. This chapter covers composing the space before you go live: the surfaces (screens) your feeds land on, the feeds themselves, cameras…

02 · 2. Setting Up a Venue

For the **operator**. This chapter covers composing the space before you go live: the surfaces (screens) your feeds land on, the feeds themselves, cameras you publish from the operator machine, branding, and the scenes you'll drive during the show.

> **Preview.** The composer runs in early builds and is feature-complete enough to author a multi-screen, multi-scene venue. It has been verified at the build level but has **not yet had a full interactive sign-off pass**, and the finished venue **art** is still being made — see the status notes throughout and in Chapter 6.

Becoming the operator

By default, opening a world puts you in as an **attendee** (world view only). To get the authoring tools you switch to the **operator** role:

- **Native desktop app** — the **Producer Controls** window (open it from the Window menu, or the ⌘⇧P shortcut) is your operator console. The main window is the world/attendee view, with an **Operator console** button on its heads-up display to unlock authoring inline.
- **Browser** — open the operator variant of the URL (the composer and producer panel appear) rather than the plain attendee link.

The backend rejects operator actions from attendees, so the role is a real boundary, not a UI hint.

Once you're operator you have two modes:

- **Compose** — design the room (this chapter).
- **Run** — drive the live show (the producer console).

The "show file": what a venue is made of

A venue is a small set of building blocks you arrange. Think of them as the parts of a real production:

BLOCK	WHAT IT IS	REAL-WORLD ANALOGUE
Venue	The space itself — geometry, floor, spawn/seating points, breakout areas	The building
Surfaces	Screens and walls your feeds play on — main screen, IMAG side walls, banners	The video walls
Feeds	Named sources of picture — a live camera room, a file, an image, a slide deck	Your inputs on a switcher

BLOCK	WHAT IT IS	REAL-WORLD ANALOGUE
Routes	Which feed shows on which surface	Patching a source to a screen
Lights	The stage and house lighting rig	The lighting rig
Zones	Named areas with capacity and access (stage, floor, VIP, backstage)	Roped-off areas
Scenes	Saved looks — a camera framing + lighting mood + what's on each screen	Show "states" you cut between
Branding	Logo, colour palette, signage, welcome copy	The event's look

You compose these before the event and **publish**. Publishing is the commit step — it saves the arrangement so every client (including people who join late) loads the same room.

> **On the venue itself.** Today the venue is a functional **graybox** — a real stage, screen, floor, and breakout zones you can compose against. Dropping in a finished, art-directed venue model (a GLTF/GLB, with baked lighting and detailed geometry) is supported by the engine but the **art is still being made**. `in development` Volumetric ("Gaussian splat") venues captured from the real world are an early scaffold. `planned`

Placing and arranging surfaces

Surfaces are the screens. The main stage screen exists by default; you add side screens (IMAG walls), banners, and presenter tiles as needed.

In **Compose** mode you select an object and move, rotate, and resize it with 3D transform handles (gizmos):

- **Select** a surface to attach the handles to it.
- **Move / rotate / scale** with the gizmo.
- ⬇ **Drop to ground** seats the object neatly on the floor.
- ⌞ **Snap to surface** aligns it flat against the nearest wall or surface it faces — handy for hanging a screen on a wall without fiddling.

Two conveniences make this readable while composing:

- **Helper markers** show non-visible things — spotlight cones, house-light bulbs, and spawn-point arrows — so you can see where lights point and which way attendees will face when they arrive. These helpers only appear in Compose mode.
- **Live collaborative editing** — when you drag an object, the move streams in real time to every other connected client (a second operator screen, an attendee view, a headset) instead of jumping only when you publish. (preview — build-verified; the multi-client live-drag has not yet had an interactive sign-off.)

Multi-presenter layouts

For panels and multi-camera talks, the composer offers ready-made **presenter layouts** that place several tiles facing the audience:

- **Grid 2×2** — four equal tiles.
- **PiP** — one big picture with a smaller inset.
- **Speaker + thumbnails** — one large speaker with a row of smaller feeds.

Each tile is its own surface, so you route a different live feed onto each one (see Routing feeds below). If a single room has several people publishing video, the world can surface each publisher as its own feed and bind each one to a distinct tile. (preview — verified against live LiveKit; you need two or more publishers in the room to see all tiles populate.)

Feeds: naming your sources

A **feed** is a named source of picture. You define feeds in the composer's **Feeds** list. A feed is one of:

- **Live (LiveKit room)** — a live camera or switcher, identified by a room name (e.g. the reserved `ace-stage` for your program mix, or `ace-cam-2` for an extra angle). This is the main path — see Chapter 3.
- **Image** — a still.
- **Slide deck** — a folder of pre-rendered slide images you step through with ◀ ▶ during the show. Only the current slide index travels to attendees, so it stays crisp and light. A laser-pointer dot you paint on the slide can be streamed to everyone too. (preview — decks are pre-rendered image folders today; in-app PDF/PowerPoint conversion is planned.)
- **File** — a video file feed (useful for a pre-roll or a fallback while you set up).

To add a live feed: **Compose ▸ Feeds ▸ "Live (LiveKit room)"**, give it a label (e.g. "Stage cam"), set its room name, and add it.

Routing feeds onto surfaces

A **route** binds one feed to one surface — "show this on that screen." With your surfaces placed and your feeds named:

1. In the producer panel, open **Feed routing**.
2. Point a surface at a feed.

The main program feed (room `ace-stage`) lands on your main screen and any IMAG side walls; iso cameras go on side surfaces; presenter feeds go on the layout tiles. You can re-route live during the show from the same place — see Going Live.

Publishing a camera from the operator machine

You don't always need OBS. The composer can turn the **operator's own webcam or a screen/window share** into a feed — good for a quick extra angle or to share a slide from your desktop:

1. **Compose ▸ "Publish a camera (this device)"**.
2. Pick **Webcam** or **Screen / window**, choose a room name (e.g. `ace-cam-3`) and a feed label.
3. ● **Start publishing** — this publishes the device into the room **and** auto-adds a matching live feed.
4. Route it onto a surface as above. ■ **Stop publishing** ends it.

This uses a publish-grant token (for camera/screen), which the app obtains for you. Full details on all the ways to get video in are in Chapter 3.

Branding

Set the event's look in the composer's **branding**: your **logo**, a **colour palette**, **signage**, and the **welcome copy** attendees see when they arrive. The Intro scene shows a greeter banner, and avatars carry billboarded nameplates so people can tell who's who.

Scenes: saving looks to cut between

A **scene** is a saved combination of three things:

- **Camera framing** — where the director camera sits and what it frames (e.g. wide on the stage, or in among the crowd).
- **Lighting mood** — the rig's look for that moment (house up, stage key, accent spots).
- **Per-surface content** — what's on each screen.

Out of the box the world ships three scenes as a starting point:

- **Intro** — a welcoming look with the greeter banner, for as people arrive.
- **Main stage** — the show look, framed on the stage.
- **Breakout** — a look oriented toward the breakout areas.

You cut between scenes during the show from the producer console, and the change applies to **everyone at once**. Authoring your own scenes (custom camera framings, moods, and content per surface) builds on these presets. Lighting driven by a **real** lighting desk (DMX / Art-Net) so the virtual rig moves with the physical rig is planned.

Zones

Zones are named areas — stage, floor, breakout, VIP, backstage — each with a capacity and an access rule. During the show you **lock and unlock** them to shape where people can be (open the breakout

rooms after the keynote; close backstage). Lock state and live occupancy sync to every client. You manage zones live from the producer console — see Going Live ▸ Zones.

> **Audio zones** — anchoring different audio to different areas (a quiet lobby, a loud main floor) is designed but the geometry model for audio-specific zones is not built yet. `planned`

Publish, and you're ready

When the room looks right, **publish** the venue. That saves the arrangement so every attendee — including late arrivals — loads the same space. From here you're ready to feed the stage (Chapter 3) and go live (Chapter 4).

SECTION 03

3. Feeding the Stage

For the operator. This chapter is about Ingest — getting live video (and the show audio) into the world. There are four ways to bring a camera in; pic…

03 · 3. Feeding the Stage

For the **operator**. This chapter is about **Ingest** — getting live video (and the show audio) into the world. There are four ways to bring a camera in; pick per source. It also covers how audio flows and where ACE Presenter is headed.

> **Preview — verified.** The main path (OBS / vMix → LiveKit → the stage screen) has been confirmed end-to-end on real LiveKit Cloud: a live publisher's track appeared on the stage screen, sub-second. The iso-camera, encoder, and in-browser paths use the same machinery. Pointing a full **production switcher via WHIP at the reserved room** is the intended live setup; the last mile (a real OBS box replacing the test publisher) is being finalised.

The model in one line

`switcher / camera ⟶ LiveKit room ⟶ a "feed" ⟶ routed onto a surface (screen / wall)`

- Your program output goes to **one reserved room,** `ace-stage` — the **Program (PGM)** feed. Route it onto the main screen and any IMAG side walls.
- Each **extra camera / angle** publishes into **its own** room and becomes an additional live feed you route to a side surface.
- **Audio** rides the program room and plays as a flat venue **PA** (full level everywhere). Attendee **voice** is separate and spatial — see Audio.

Transport is one WebRTC stack (LiveKit), so video, voice, and spatial audio all travel together — you don't run separate systems for each.

Prerequisite: a LiveKit project

Live feeds ride on **LiveKit** (a WebRTC media server). In the current preview an operator uses a **LiveKit Cloud** project (free tier is fine) and points their encoder at it. In the released product this is intended to be brokered through the ACE gateway on your license, so you won't handle raw keys — that pooled path is wired but still being finished. `in development`

If no LiveKit is configured, the world simply stays on its **file feed** (a bundled or supplied video) — nothing breaks, you just don't get live.

Four ways to get a camera in

1. OBS / vMix → PGM (the main feed) `preview · verified`

This is the primary, verified path. In OBS:

1. **Settings ▸ Stream ▸ Service: WHIP**.
2. Set the **URL** and **bearer token** from your LiveKit project's ingress for room `ace-stage`.
3. **Start Streaming**.

The **Program** feed lights up the main screen, and the audio in this stream becomes the venue PA. vMix works the same way via its WHIP/external output.

2. OBS / vMix → an iso room (an extra angle)

Run a **second** WHIP output — a second OBS instance, or a vMix external output — into its **own** room, for example `ace-cam-2`. Then in the composer:

1. **Compose ▸ Feeds ▸ "Live (LiveKit room)"** — label it (e.g. "Stage cam"), set the room to `ace-cam-2`, and **+ Add feed**.
2. In the producer panel, open **Feed routing** and point a **side surface** at it.

> Iso cameras are **video-only** to the world. The show audio always comes from the program room, so the world ignores iso-room audio to avoid doubling it.

3. Hardware encoder (RTMP / WHIP)

For pro cameras through an encoder/decoder, configure a **LiveKit ingress** for the target room (**WHIP**, or **RTMP** if your encoder only speaks RTMP), then add a **live feed** for that room exactly as in path 2. Most of the work is on the LiveKit side; the world side is identical to any other live feed.

Setting up an ingress once, per room: in your LiveKit project, **Ingress ▸ Create**, choose **WHIP** (or RTMP), set the room name and a participant identity, and copy the ingress URL + key into your encoder or OBS.

4. In-browser publisher (no OBS) `preview`

Turn the **operator's own webcam or a screen/window** into a camera — the quickest possible angle, no external software:

1. **Compose ▸ "Publish a camera (this device)"** → pick **Webcam** or **Screen / window**, set a room (e.g. `ace-cam-3`) and a feed label.
2. ● **Start publishing** — this publishes the device into the room and auto-adds a matching live feed.
3. Route it onto a surface. ■ **Stop publishing** ends it.

Publishing needs a publish-grant token (distinct from the attendee mic-only token). The app fetches it for you.

How audio flows

There are **two independent audio buses**, and the world sorts sources into them automatically:

BUS	SOURCE	BEHAVIOUR	HOW IT'S IDENTIFIED
Program	The PGM room's audio — your OBS/switcher mix	**Flat PA** — same level everywhere, no distance falloff	A participant that publishes **video** is treated as program
Voice	Attendee microphones	**Spatial** — positioned at each speaker's avatar, quieter with distance	An **audio-only** participant is treated as attendee voice

So you don't tag anything: OBS (which publishes video) becomes the PA, and a phone attendee (audio only) becomes a positioned voice in the crowd. You balance the two buses, mute the crowd, or mute one person from the producer console — see Going Live ‣ Audio mix. A synthetic room **reverb** gives voices a sense of the space; swapping in a recorded impulse response of a real venue for higher fidelity is supported. (preview — spatial voice is built and running; a session with several mic'd people is needed to sign it off perceptually.)

A note on stage videos (file feeds)

When you use a **file** feed (a pre-roll, or a fallback while you set up), the video must be a **faststart MP4** — one whose index is at the front of the file — or browsers won't stream it smoothly. If you prepare video for this, export or remux it as faststart. Live feeds don't have this constraint; it only applies to file feeds.

Where ACE Presenter fits

The intended suite handoff is that **ACE Presenter publishes straight into a world surface** — either as a **video** feed (its program output → LiveKit) or as **structured data** (slides and lyrics sent over a data channel and rendered natively in-world, crisp at any distance), with cues syncing slide changes. That would make the world just another output ACE Presenter already speaks to, in the same cueing language as the rest of the suite.

This direct integration is **planned**, not built. **Today**, you bridge them the same way as any other source: send ACE Presenter's output through **OBS/vMix (or an NDI/capture bridge) → LiveKit** as a normal live feed. `planned`

Quick reference

WANT	DO
Main screen shows the show	OBS **WHIP →** `ace-stage`; route **Program** onto the main surface

WANT	DO
A second camera angle	Publish to its **own** room (path 2/3/4) → add a **live feed** → route to a side surface
Show audio fills the venue	Automatic — the program room's audio plays as flat PA
A quick angle with no OBS	**Compose ▸ Publish a camera (this device)** → Webcam or Screen
No live feed available	The world stays on the file feed — nothing breaks

SECTION 04

4. Going Live & the Producer Console

For the operator. Your venue is composed (Chapter 2) and your feed is ready (Chapter 3). This chapter is running the show: going live, sharing the lin…

04 · 4. Going Live & the Producer Console

For the **operator**. Your venue is composed (Chapter 2) and your feed is ready (Chapter 3). This chapter is running the show: going live, sharing the link, and driving the room from the producer console — scenes, cues, lighting, the audio mix, zones, and moderation.

> **Preview.** The producer console runs in early builds and drives scenes, cues, lighting moods, feed routing, audio, and zones — all synced to every client. It is verified at the build level and against live LiveKit; a full interactive show sign-off is still to come. **Frame-accurate** alignment of the feed with cues across every screen (temporal sync) is in development.

Going live

The ● **Go live** control lives on the operator console. Pressing it:

1. **Creates the event** as its own isolated world, seeded with the venue you composed.
2. Binds you (the operator) into the event's room.
3. Shows you a **join code** (a short code like `3XH4H`) and a **copyable link** (a URL with the code in it).

Then:

1. **Share the link** with your audience — email, chat, a page, however you reach them. That's all they need; no account, no install (see Chapter 5).
2. **Start your stream** (OBS/vMix), and switch the stage feed to **live** — the stage screen lights up with your program.

● **End event** stops new joins and closes the world. Multiple operators can run **concurrent** events; each is a fully separate world with its own people, scene, feeds, and audio mix.

> **Metering.** Going live is intended to register the event against your ACE **license** and meter the audience joins. That production billing path is wired end-to-end but rides on account infrastructure still being finished — in the current preview the link works against the world backend regardless. See Status. (in development)

The producer console

On the **native desktop app** the producer console is its own **always-on-top window** (open it from the Window menu, or the ⌘⇧P shortcut), separate from the world view — so you can watch the room and drive it at the same time. In the **browser**, the producer panel appears alongside the world when you're in the operator role.

The console is organised around the things you change live: **scenes**, the **cue list**, **feed routing**, the **audio mix**, **zones**, and the **in-room** people list.

Scenes: cutting the room's look

A **scene** bundles a camera framing, a lighting mood, and what's on each screen (see Chapter 2 ▸ Scenes). Firing a scene applies it to **everyone at once**. The three built-in scenes give you a working show out of the box:

- **Intro** — welcoming look with the greeter banner, for as people arrive.
- **Main stage** — the show look, framed on the stage.
- **Breakout** — oriented toward the breakout areas.

Cut to **Intro** as the room fills, to **Main stage** when the show starts, and to **Breakout** to send people into smaller groups. Each cut changes camera framing, lighting mood, and screen content together, synced to every attendee.

The cue list

The **cue list** fires individual actions on demand — the moment-to-moment beats of your show. Cues run **backend-resolved actions** (the server decides and applies the effect), so firing a cue changes the world for every client consistently, not just on your screen. Use cues for the granular moves between scene changes — a specific lighting hit, opening a zone, a content change on one surface.

> **Timing.** Cues apply live to everyone, but **frame-exact** alignment with the video (so a lower-third lands on the same frame for every attendee) is the **temporal sync** work still in development. For now, treat cues as "apply now to everyone," which is right for the great majority of moments.

Re-routing feeds live

Feed routing in the console is live: point any surface at a different feed mid-show. Cut the main screen from the program mix to a slide deck; move an iso camera onto a side wall; swap a presenter tile. This is the same routing you set up while composing (Chapter 2 ▸ Routing feeds), available during the show.

For **slide decks**, step through with ◀ ▶ — only the current slide index travels to attendees, so it stays sharp. You can toggle a ● **Laser** pointer and hover a screen to paint a dot every attendee sees. `preview`

Mixing audio

The console's **Audio mix** section controls the two audio buses — the show's **Program** PA and the crowd's spatial **Voice**:

- **Program / Voices faders** — set the room balance for everyone. Bring voices down under a talk; bring them up for a Q&A.
- **Mute all voices (keynote)** — silences every attendee mic in one tap, so only the show audio plays. Each client drops its own mic at the source, and the local voice bus is dropped as a backstop, so the mute is real, not just lowered. Tap again to un-mute.
- **Per-attendee mute** — the 🎤 button next to a name in the **In-room** list mutes one person (moderation).

> Mutes are enforced **at the source**: a muted client turns its own microphone off. That means a mute genuinely stops the audio rather than only turning it down on other people's machines.

Managing zones

Zones are the named areas you defined while composing — stage, floor, breakout, VIP, backstage. During the show you **lock and unlock** them to shape where people can be:

- Open the **breakout** zones after the keynote to send people into groups.
- Keep **backstage** or **VIP** locked to everyone but the right people.

Lock state and live occupancy sync to every client, so what you see in the console matches what attendees experience. (Anchoring different **audio** to different zones is planned.)

Moderating the room

The **In-room** list is your roster and your moderation surface:

- See who's present.
- **Mute one person** with the 🎤 button next to their name.
- **Mute the whole crowd** with **Mute all voices (keynote)** for talks and presentations.

Because voice is spatial, disruptive audio is naturally localised — but the per-person and global mutes are there for real control when you need it.

A simple run sheet

A minimal show, start to finish:

1. **Compose** or open your venue and **publish** it. → Chapter 2

2. Have **OBS/vMix ready** to stream to `ace-stage`. → Chapter 3
3. ● **Go live**; copy the **link**; share it.
4. Cut to the **Intro** scene as people arrive.
5. **Start the stream**; switch the stage feed to **live**.
6. Cut to **Main stage**; run the show with **scenes** and **cues**.
7. Balance the **audio mix**; **mute the crowd** for talks, un-mute for Q&A.
8. Open **breakout** zones for small groups if you want them.
9. ● **End event** when you're done.

What's still being built here

To keep expectations honest, the show-control features not yet finished:

- **Temporal (frame-accurate) A/V sync** — aligning the feed and cues to the exact same frame for every attendee. (in development)
- **Live captions** of the stage audio via speech-to-text. A local-microphone demo caption source exists; the production stage-audio caption source is `in development`.
- **Real lighting-desk ingest** (DMX / Art-Net) so the virtual rig mirrors a physical rig. `planned`
- **Audio zones** — different audio anchored to different areas. `planned`

See Chapter 6 for the full board.

SECTION 05

5. Joining as an Attendee

For the attendee — the person who was invited. If someone sent you a link to an ACE Virtual World event, this is the only chapter you need. No account...

05 · 5. Joining as an Attendee

For the **attendee** — the person who was invited. If someone sent you a link to an ACE Virtual World event, this is the only chapter you need. **No account, no app to install, no license.** You open a link and you're in.

> **Preview.** Joining by link, watching the stage, moving around, and talking in spatial voice all work in early builds. The venue and avatars are currently clean **placeholder art** while the finished 3D art is being made — everything works, it just isn't the final look yet.

What you need

- A **modern browser** — Chrome or Edge are the smoothest for the 3D and voice; recent Safari and Firefox work too. On a desktop, laptop, or phone.
- A **microphone** if you want to talk (optional — you can just watch).
- Optionally, a **Meta Quest** headset to step inside in full VR (see In a Quest headset).

That's it. You do **not** need an ACE account or any download.

Joining in a browser

1. **Open the link** you were given. It looks like `https://world.ace-presenter.app/?code=XXXXX` — the short code is built in.
    - If you were given a bare **join code** instead of a full link, open the event page and enter the code when asked.
2. The world loads. Click **Enter world** (and, if prompted, allow **microphone** access so you can talk — you can decline and still watch).
3. You're in, standing in the venue as an **avatar**, with the stage screen in front of you.

If the operator hasn't gone live yet, or has ended the event, the link will tell you — hold tight, or check back at the time you were given.

Looking around and moving

You explore the space like a simple 3D game:

- **Look around** — drag with the mouse (or move your finger on a touchscreen) to turn your view.

- **Move** — walk toward the stage or over to a breakout area. On a keyboard, the movement keys move you; on touch, use the on-screen controls.
- **Jump to a seat** — you can hop between seating spots to find a good view rather than walking the whole way.

You'll see **other attendees as avatars** with **name tags** floating above them, so you can tell who's who. As the show runs, the **producer** may change the scene — the camera framing, the lighting, and what's on the screens will shift for everyone at once. That's normal; just enjoy it.

Talking: spatial voice

Voice in ACE Virtual World is **spatial** — it comes from the direction of whoever's speaking and gets quieter with distance, so the room sounds like a real room:

- **Walk toward someone** to hear them more clearly; drift away and they fade.
- Turn your head and voices shift around you.
- To talk, make sure you allowed **microphone** access when you joined. If you skipped it, you can grant it in your browser's site settings and rejoin.

The **show audio** (the stage feed's sound) is different — it plays at a **full, even level everywhere**, like a PA system, so you always hear the presentation clearly no matter where you stand.

A few things the producer controls: they can turn **all** attendee voices down or off during a talk or keynote (you'll still hear the show), and they can mute **one** person for moderation. If your mic goes quiet during a presentation, that's usually the producer holding the room for the speaker — it'll come back for discussion and Q&A.

In a Quest headset

You can step inside the event in full VR on a **Meta Quest**:

1. Open the **same link** in the **Meta Quest browser**.
2. Choose **Enter XR**.
3. You're in the venue in immersive VR — look around by moving your head, and hear voices positioned around you in 3D.

The browser and the headset join the **same event** — some of your group can be on laptops while others are in headsets, together in the same room. Full VR hand and locomotion polish and an in-headset UI are still being refined; the core experience — being present, watching, and hearing the room — works today.

> **Installable.** The attendee experience can also be installed as a lightweight app (a PWA) from the browser, so repeat visits load fast. This is optional — the plain link always works.

Troubleshooting

PROBLEM	TRY THIS
Link says the event isn't live	The operator hasn't gone live yet or has ended it. Check the time you were given.
No sound / can't hear the show	Check your device volume and that the browser tab isn't muted; some browsers need a click before audio starts — click into the world.
Others can't hear me	Make sure you allowed **microphone** access; check your browser's site permissions, then rejoin. Also, the producer may have muted the crowd for a talk.
The 3D is choppy	Close other heavy tabs/apps; use Chrome or Edge; on a phone, moving somewhere with a stronger connection helps.
Nothing loads	Refresh the page. If it still fails, your network may be blocking real-time media — try a different network.

Your privacy, briefly

You join **without an account** and **without signing in**. You appear to others as an avatar with the name you're given or choose, and — if you turn your mic on — your voice. That's the extent of it. If you'd rather just watch, don't enable your microphone and you'll be a silent presence in the room.

SECTION 06

6. Platforms, Requirements & Status

The candid chapter. What runs ACE Virtual World, what the network needs, and an honest, layer-by-layer board of what works today versus what's still b...

06 · 6. Platforms, Requirements & Status

The candid chapter. What runs ACE Virtual World, what the network needs, and an honest, layer-by-layer board of what works today versus what's still being built. Because this is a pre-release product, this page is the authoritative status reference for the whole guide.

Platforms

WHO	PLATFORM	HOW IT RUNS
Operator	Native desktop app (macOS / Windows via Electron)	The world view plus a separate always-on-top **Producer Controls** window for the console
Operator	Modern browser	The same world, with the composer and producer panel in the operator role
Attendee	Modern browser (desktop / laptop / phone)	Zero-install; open the link and **Enter world**
Attendee	**Meta Quest** headset (WebXR)	The same link in the Quest browser → **Enter XR** for immersive VR
Attendee	Installable app (PWA)	The browser experience installed for fast repeat visits

Browser recommendation: Chrome or Edge give the best WebGL and WebRTC behaviour. Recent Safari and Firefox work for attendees.

Requirements

For attendees

- A modern browser and a reasonable internet connection.
- A **microphone** to talk (optional — watching needs nothing extra).
- For VR: a **Meta Quest** headset.
- **Nothing else** — no account, no install, no license.

For operators

- The desktop app or a modern browser, and enough machine to run a 3D scene comfortably (a discrete GPU helps for large crowds).
- A way to get video in — usually **OBS or vMix** — pointed at a **LiveKit** project over **WHIP** (see Chapter 3).

- An ACE **license** for the operator/producer role (the audience is always free). The licensed, metered go-live path is still being finished — see below.

Network & hosting

Because the audience reaches the show over the internet, the real-time services must be **hosted and reachable** — they can't live only on the operator's laptop behind a home router:

- **LiveKit** (the video + spatial-audio transport) is hosted (LiveKit Cloud). ✅
- **The world backend** (presence, producer state, feeds, config) must be deployed somewhere public and multi-tenant, so one central deployment can host many operators' events, each fully isolated from the others. ✅ (multi-event isolation is in place)

Real-time media (WebRTC) needs UDP to flow; on very locked-down corporate or school networks it can be blocked. If an attendee can't connect at all, a different network usually fixes it.

Status board

The heart of this chapter. Read it by the five layers. Badge meanings are in the README: (preview — verified), `preview`, `in development`, `planned`.

Layer 1 — Ingest (getting content in)

FEATURE	STATUS	NOTES
OBS / vMix → LiveKit (WHIP) → stage screen	`preview · verified`	Confirmed end-to-end on real LiveKit Cloud; the primary path
Iso / extra cameras (own room → side surface)	`preview`	Same machinery as the main feed; video-only to the world
Hardware encoder (WHIP / RTMP ingress)	`preview`	Mostly LiveKit-side config; world side identical
In-browser publisher (operator webcam / screen)	`preview`	Publishes the operator's device and auto-adds a feed
Slide-deck feed (pre-rendered images) + laser pointer	`preview`	Only the slide index travels; in-app PDF/PPTX conversion is `planned`
ACE Presenter → world (video and/or structured data)	`planned`	Bridge via OBS/NDI today; native suite handoff is designed, not built

Layer 2 — Transport (real-time delivery)

FEATURE	STATUS	NOTES
One WebRTC stack (LiveKit) for video + voice + spatial audio	`preview · verified`	Live video path confirmed on real keys

FEATURE	STATUS	NOTES
Multi-track ingest (N publishers in one room → N feeds)	preview	Verified against live LiveKit; needs 2+ publishers to populate all tiles
Pooled, license-gated media tokens via the ACE gateway	in development	Wired; rides on account infrastructure being finished. Today the world backend mints tokens itself

Layer 3 — World (the venue)

FEATURE	STATUS	NOTES
Composable graybox venue — stage, screens, floor, breakout	preview	Real geometry you compose against
Composer — surfaces, feeds, routing, gizmos, drop/snap, branding, scenes	preview	Build-verified; a full interactive sign-off pass is still to come
Live collaborative editing (drags stream in real time)	preview	Build-verified; multi-client live-drag not yet interactively signed off
Multi-surface routing + presenter layouts (Grid / PiP / Speaker)	preview	Each tile is a surface you route a feed onto
Finished **venue art (GLTF)** — detailed geometry, baked lighting	in development	Engine consumes it; the art itself is being made
Lightmap-baked lighting on a real venue model	in development	Loader is built; the bake is external art-pipeline work
Stage floor reflections / image-based lighting	preview	Runs; needs visual tuning on a display
Volumetric ("Gaussian splat") venues from real-world captures	planned	Early scaffold only

Layer 4 — Presence (the people)

FEATURE	STATUS	NOTES
Avatars, movement, name tags	preview	Capsule placeholders until finished art loads
Spatial voice (positioned per avatar, distance falloff)	preview	Built and running; needs a several-person mic'd session to sign off perceptually
Room reverb (sense of the space)	preview	Synthetic default ships; a recorded venue impulse response can be swapped in
Crowd scaling (LOD, interest management for hundreds)	preview	Built; runtime-confirmed via a crowd simulator
Finished **avatar art (GLB)** + lip-sync (mouth movement)	in development	Code is in place; needs the rigged art dropped in

FEATURE	STATUS	NOTES
Full VR hand / locomotion polish + in-headset UI	planned	Core VR presence works; refinement pending

Layer 5 — Production (show control)

FEATURE	STATUS	NOTES
Go live / end event; join code + shareable link	preview	Creates an isolated per-event world
Scenes (Intro / Main / Breakout) — camera + lighting + content	preview	Synced to every client
Cue list — backend-resolved actions fired to everyone	preview	
Feed routing, live	preview	Re-route any surface mid-show
Audio mix — Program/Voice faders, mute-all, per-attendee mute	preview	Mutes enforced at the source
Zone lock/unlock with live occupancy	preview	
Temporal (frame-accurate) A/V sync	in development	The hardest item; cues apply "now to everyone" until it lands
Live captions of stage audio (speech-to-text)	in development	A local-mic demo source exists; production stage caption source pending
Metered go-live on your license (production billing)	in development	End-to-end path is wired; rides on account infrastructure being finished
Real lighting-desk ingest (DMX / Art-Net)	planned	Virtual rig mirroring a physical rig
Audio zones (different audio per area)	planned	Needs an audio-zone geometry model

The one big caveat

Most of the preview items above have been verified at the **build and type level** and, for media, **against real LiveKit** — but a full **interactive, in-headset, multi-participant show** has not yet been signed off end to end. The live media path and the crowd simulation are the parts confirmed at runtime. Treat everything here as a **preview of a product in active development**: it works in early builds, it will change, and it is not yet released.

To follow along or be brought in as it opens up, register interest.

See also

- Getting Started & Concepts — the five layers and roles
- Setting Up a Venue — the composer
- Feeding the Stage — ingest paths and audio
- Going Live & the Producer Console — running the show
- Joining as an Attendee — the audience experience