

Notifications & Buttons

Setting up the Monday email digest, instant webhook alerts, and physical Stream Deck buttons — everything that reaches the team outside the dashboard.

VERSION

1.1.0 Beta 52

UPDATED

August 8, 2026

REQUIRES

Admin access

WHAT'S INSIDE

1

The three channels

What reaches whom, and when

3

Instant webhooks

Slack, Discord, and Teams alerts

2

Monday email digest

SMTP setup for Gmail, 365, or SendGrid

4

Stream Deck buttons

Physical buttons via Bitfocus Companion, signage control, standby feedback

1 The three channels

- i **New in Beta 39 — a fourth thing to glance at, and it needs no setup at all.** The dashboard's own top bar now carries the state of the service in a word and a colour: **ALERT**, **WARNING**, **LIVE** or **STANDBY**, worst first. It is not a notification channel — nothing is sent anywhere and nobody has to configure it — but it is the fastest read in the building for "is something wrong right now", and it is on every dashboard page. Read the word rather than the colour; the word is there because green and amber are near-identical to a colourblind operator. Wall and kiosk screens deliberately never show it.

CHANNEL	WHEN IT FIRES	BEST FOR
Email digest	Every Monday morning, automatically	The whole team's weekly review — length, variance, SPL, streaming, attendance, incidents, clip counts for every service, with the leadership PDFs attached
Webhooks	The moment something happens	Devices staying down (with diagnosis), recoveries, critical alerts, service start/end — into a Slack/Discord/Teams channel
Stream Deck buttons	When someone presses them	Logging incidents, marking sermon clips, advancing the plan, and stage messages without touching a browser

2 Monday email digest

Admin → **Alerts & Notifications** → **Weekly email digest**. The dashboard has its own built-in mail client — no plugins, works with any SMTP provider.

› Gmail / Google Workspace

- 1 On the sending account: Google Account → Security → turn on 2-Step Verification if it isn't already.
- 2 Security → App passwords → create one named TechArts Dashboard. Copy the 16-character password.
- 3 In the dashboard: host `smtp.gmail.com`, port 465, implicit TLS on, username = the Gmail address, password = the App Password.

› Microsoft 365

Host `smtp.office365.com`, port 587, implicit TLS off (the dashboard upgrades with STARTTLS automatically), sign in with the mailbox credentials. If your tenant blocks SMTP AUTH, ask IT to enable it for this one mailbox.

› SendGrid / Mailgun (no church mailbox needed)

Host `smtp.sendgrid.net`, port 465, username `apikey`, password = the API key. Set From to a verified sender.

› Finish up

- 1 Set From, the To list (comma-separated), and the send hour (default 7 — Mondays, right after the checklist reset).
- 2 Press **Send test email**, then **Preview digest** to see this week's version in the browser, then check Enabled and Push live.

✓ **Send this week's digest now** exists for the Mondays you can't wait — it emails immediately and still counts as that week's send. The digest attaches the SPL-compliance and streaming-SLA PDFs (toggle in the same tab); clip-marker email to the editing team has its own card below it.

3 Instant webhooks

Admin → Alerts & Notifications → Webhook. One URL, five event toggles, one test button.

› Slack

- 1 In Slack: create a channel like #tech-arts-alerts.
- 2 slack.com/apps → search Incoming WebHooks → Add to Slack → pick the channel → copy the webhook URL (https://hooks.slack.com/services/...).
- 3 Paste it into the dashboard, choose which events post, press Send test message.

› Discord

Channel settings → Integrations → Webhooks → New Webhook → copy the URL and add **/slack to the end** — Discord then accepts the same format.

› Microsoft Teams

Channel → ... → Connectors → Incoming Webhook → configure and copy the URL.

EVENT TOGGLE	WHAT POSTS
Device offline / online	● when a device has stayed down past the alert window, with the likely cause · ● when it recovers, with downtime
Auto-diagnosis results	✂ the diagnosis detail (device vs switch vs subnet vs cabling)
Critical alerts	Anything severity-critical from the alert engine
Service start / end	🕒 PCO LIVE started the clock · ✅ report archived
Countdown milestones	T-30 / T-15 / T-5 / T-2 pings as each fires (editable in Admin → General)

- i **Tuning the noise:** the device down-window (Admin → Alerts → "after it has been down for") is how long a device must be down before anything fires — default 60 seconds. Brief blips never reach the channel, and when the device comes back its alert acknowledges itself on the dashboards.

4 Stream Deck buttons (Bitfocus Companion)

- 1 Install Bitfocus Companion (free, bitfocus.io) on any machine with the Stream Deck attached — the booth PC is perfect.
- 2 In Companion, add a button → action Generic: HTTP Request → method GET → paste a URL from the table below (they're also listed, pre-filled with your key and with copy buttons, in Admin → Alerts & Notifications → Stream Deck / Companion control API).
- 3 Repeat per button. Suggested top row: Mic dropout, Slide glitch, Stream stutter, 🎤 Mark clip, Next item.

BUTTON	URL (REPLACE HOST AND KEY)
Log incident	<code>http://10.0.5.20:4400/api/control/incident?key=KEY&tag=Mic dropout</code>
Mark sermon clip	<code>.../api/control/mark-clip?key=KEY&label=Key point</code>
Next plan item	<code>.../api/control/next-item?key=KEY</code>
Stage message	<code>.../api/control/stage-message?key=KEY&text=Wrap it up</code>
Clear stage message	<code>.../api/control/stage-clear?key=KEY</code>
Acknowledge alerts	<code>.../api/control/ack-alerts?key=KEY</code>
Log attendance count	<code>.../api/control/attendance?key=KEY&count=312&note=Main auditorium</code>
Flag gear for repair	<code>.../api/control/flag-repair?key=KEY&gear=Mic 3&title=Crackling&priority=urgent</code>
Countdown → stage	<code>.../api/control/countdown-stage?key=KEY</code>
Signage — next page	<code>.../api/control/signage-next?key=KEY&screen=lobby</code>
Signage — previous page	<code>.../api/control/signage-prev?key=KEY&screen=lobby</code>
Signage — hold / resume	<code>.../api/control/signage-toggle?key=KEY&screen=lobby</code>
Signage — jump to a page	<code>.../api/control/signage-page?key=KEY&screen=lobby&page=3</code>
Status (JSON)	<code>.../api/control/status?key=KEY</code>

i **New in Beta 38 — booth-panel presses show up in the Activity Log.** Eight of the buttons above are recorded in the **Activity Log** — AV Central → **Settings** → **Activity Log** since Beta 40, Admin → Activity Log before that: log incident, mark clip, next item, stage message, clear stage message, attendance, flag repair, and countdown → stage. They are recorded as **the control key**, named for the booth panel — *not* as a person, because there is nobody behind a shared button on a wall and inventing a name would be a fiction. The other five are deliberately not recorded: `status` is polled several times a minute for button feedback, and the acknowledge and signage buttons are the same actions as their on-screen equivalents, which are excluded for volume. Recording them here but not there would be incoherent.

› The four signage buttons — new in Beta 19

These drive the `/signage` wall screens: step a page, jump to a page, or hold one page on screen.

PARAMETER	WHAT IT DOES
<code>&screen=</code>	Which TV the button acts on. It is the name that TV gives itself in its own address — <code>/signage?screen=lobby</code> . Leave it off and the button drives the screen called <code>default</code> , which is any TV opened at plain <code>/signage</code> . Names are squeezed to lowercase letters, digits, <code>-</code> and <code>_</code> , so <code>Lobby TV!</code> becomes <code>lobbytv</code> — use the same plain name on the TV and in the button. Point a button at a name no TV uses and nothing appears to happen.
<code>&page=</code>	On <code>signage-page</code> only, and it counts from 1 — <code>page=1</code> is the first page in the rotation list on Admin → Screens & Displays, not the second. Omit it and the request is refused with 400 and a plain-language reason rather than silently doing nothing.
<code>&mode=</code>	On <code>signage-toggle</code> only. With nothing, it flips between holding and rotating. With <code>&mode=static</code> or <code>&mode=scroll</code> it forces one of them — better for a Companion button that should always mean "hold" (or always mean "let it run") no matter what state the screen is in.

- i **Next, previous and jump all put the screen on hold.** That is deliberate — without it the rotation timer would pull the page away a few seconds after the press, which reads as a broken button. Send `signage-toggle` (or `&mode=scroll`) to set it rotating again. The hold is stored in the dashboard's configuration, not in the browser, so it survives a refresh and a power cut: a screen you park stays parked until somebody unparks it.

› Button feedback with the status feed

The status URL is read-only JSON built for Companion feedbacks and variables. Poll it with Companion's Generic HTTP module (a 5–10 s interval is plenty) and you get: `live`, `remainSec`, `currentItem`, `nextItem`, `spl`, `viewers`, `activeAlerts`, `encoderKbps`, `encoderHealth`, and — new in Beta 19 — `signageScreens`, an object keyed by screen name giving each one's `mode` (`scroll` or `static`), its 1-based `page`, and the `kind` of page it is on. Typical uses: a key that turns red while `activeAlerts > 0`, a T-bar key showing time remaining, an encoder key that goes amber when `encoderHealth ≠ good`, and a signage key that lights while `signageScreens.lobby.mode` is `static`.

› Standby on a button — new in Beta 19

Add `&role=<position id>` to the status URL — e.g. `.../api/control/status?key=KEY&role=lighting` — and the response additionally carries that one position's standby state as flat values, so a feedback binds to a boolean and a button's text to a string with no JSON path to walk:

FIELD	WHAT IT HOLDS
<code>standbyOn</code>	<code>true</code> while that position has a cue coming, <code>false</code> otherwise. Bind a key colour to this.
<code>standbyText</code>	The operator-voice line: "Your item starts in ~2 min", "You're 2 slides out", "You're next", "Your cue is on this slide".
<code>standbyShort</code>	The same thing in button-sized form: "~2 min", "2 slides out", "next slide", "this slide".
<code>standbySlidesOut</code> · <code>standbySecondsOut</code>	The raw numbers, whichever one applies; the other is <code>null</code> .

standby

Without `&role=`, every position that has a standby, most imminent first — the whole-booth view.

! **A dark standby key is the correct state, not a broken one.** The dashboard deliberately reports nothing when it cannot prove where the service is — no service running, ProPresenter not reporting a live slide, a cue anchored to a word that isn't in this week's script, an item with no scheduled length, or the position set to *off* in Admin → Roles & Positions. In all of those `standbyOn` is `false` and `standby` is empty. A button that lights on a stale number teaches people to ignore it, so it stays dark instead. Build the button as a heads-up, never as somebody's only warning.

! **The key is the lock.** Anyone on the network with the key can press these "buttons." Change the Control key in Admin → Alerts & Notifications to revoke every old button at once, then update Companion.

This document references dashboard version 1.1.0 Beta 52. What changed in each production release lives in the separate **Version History** document — beta releases included.