maritime

USER MANUAL

Maritime Lighthouse

Pixel lighting control over sACN and Art-Net. Draw the rig, wire it, run the cues — from one Mac, a grandMA3, or a Stream Deck.

VERSION **0.4.0** · MACOS 11 OR LATER
GETSTAGE.WORK/SOFTWARE

STAGE · SCENIC · SIGNAL

CONTENTS

What's in this manual

Lighthouse is built around the order you actually work in: draw the rig, cable it, check the patch, prove the wiring, then run looks. The chapters follow that order. If you already have a rig file, jump to chapter 6. Driving Lighthouse from a lighting console or a Stream Deck is chapter 11.

CHAPTER 1

Install and first launch

Lighthouse is a single Mac app. There is no server to run, no account to create, and nothing that needs the internet once it is on the laptop.

What you need

- A Mac running macOS 11 Big Sur or later. There are separate downloads for Apple Silicon (M1 and later) and Intel.
- A pixel controller that speaks sACN (E1.31) or Art-Net, on a network the Mac can reach. Lighthouse ships configured for a CLEN CL-404R, but any controller with those protocols works.
- Pixel bars or strip on the controller's ports. The fixture library covers CLEN Pixel Bar Pro 24 V bars (1 m, 0.5 m, 0.3 m, 0.25 m and the 90° corner) and the 2D connectors that join them.

Install

1. Download the DMG for your Mac from **getstage.work/software**.
2. Open the DMG and drag **Lighthouse** onto the **Applications** shortcut.
3. Eject the DMG and launch Lighthouse from Applications or Spotlight.

Lighthouse is signed with Maritime Creative's Apple Developer ID and notarized by Apple, so it opens with a normal double-click. No "unidentified developer" detour.

> **LOCAL NETWORK PERMISSION**
>
> On first launch macOS asks whether Lighthouse may find and connect to devices on your local network. **Allow it.** Lighthouse sends every frame of lighting data over that network; if you deny the prompt, the app runs but nothing leaves the Mac. To change it later: System Settings → Privacy & Security → Local Network.

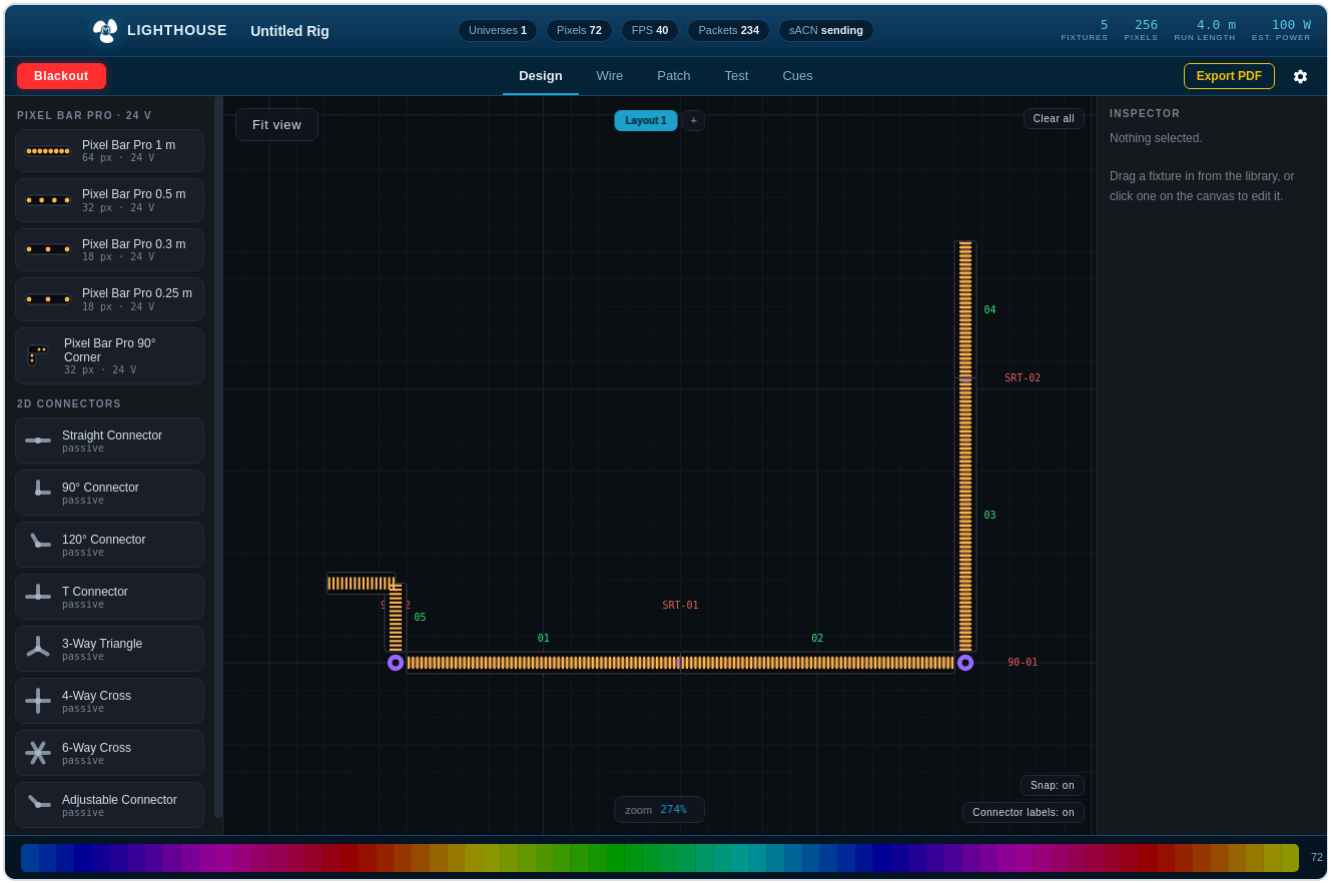
First launch

Lighthouse opens on the **Design** tab with an empty canvas and a rig called *Untitled Rig*. The engine is already running and, once anything is patched, already sending — there is no separate "go live" switch. The status pills in the title bar tell you what is going out at any moment.

The fastest way to get light out of a real rig is the order the tabs are in: draw it on **Design**, cable it on **Wire**, glance at **Patch**, pick a look on **Cues**. That is chapters 3 through 6.

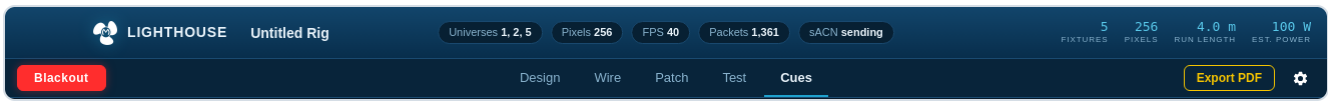
CHAPTER 2

The window



The Lighthouse window. Title bar with status pills on top, the tab strip beneath it, and the current page below. This is the Design tab with a small demo rig drawn.

Title bar



ITEM	WHAT IT TELLS YOU
Rig name	The name of the open rig. Click it to rename; the name also titles any PDF you export.
Universes	Which DMX universes are currently being output.
Pixels	Total pixels in the patch.
FPS	Measured frame rate the engine is achieving. Should sit at the rate set in Settings (40 by default).
Packets	Running count of packets sent. If this stops climbing, nothing is leaving the Mac.
sACN / Art-Net	**sending** means the socket is open and frames are going out. **not sending** in red means the socket could not open — usually the wrong network interface.
Rig stats	Fixtures, pixels, total run length and estimated power draw for what is on the canvas.

Tab strip

CONTROL	PURPOSE
Blackout	Kills all output instantly from any page. The button flashes red while blackout is engaged; press it again to restore. Also `⌘B`.
Design	Draw the physical layout — bars, corners, connectors.
Wire	Place controllers and run data cables between ports and bars.
Patch	The fixture table the engine actually uses: universe, address, pixel count, colour order.
Test	Live view of every channel in a universe, plus tools to force a fixture to a colour.
Cues	Looks, saved cues, master and speed, and a live monitor of the rig.
Export PDF	Prints the layout as a plot with a pull list and pixel patch.
⚙ Settings	Network interface, sACN, Art-Net, frame rate, control API, rig file.

> **DESIGN AND WIRE ARE ONE DRAWING**
>
> They are two views of the same canvas. Design shows the light and hides the controllers; Wire shows the controllers and the data run. Anything you draw on one is there on the other.

Keyboard shortcuts

APP

⌘B	Blackout on / off
⌘N	New rig
⌘O	Open rig…
⌘S	Save rig
⇧⌘S	Save rig as…
⌘L	Undock the live visualizer into its own window

CANVAS (DESIGN / WIRE)

F	Fit view (on Wire with a bar selected: flip its ends)
S	Toggle grid snap
Space + drag	Pan
Scroll	Zoom around the cursor
R	Rotate selection 90°
D	Duplicate selection
⌫	Delete selection
⌘Z / ⇧⌘Z	Undo / redo
Arrow keys	Nudge selection

CHAPTER 3

Design — draw the rig

Design is a top-down drawing of the rig at real scale. Bars snap into connectors the way the hardware does, so what you draw is what gets built — and later, what gets patched.

The canvas

The grid is in centimetres; one square is 10 cm. **Fit view** (top left, or `F`) frames everything you have drawn. The zoom readout sits at the bottom centre. **Snap** and **Connector labels** toggles are bottom right; **Clear all** is top right and asks you to type `yes` before it does anything.

Every tab keeps its own framing: switching to Design always shows you the whole layout, switching to Wire always shows you the whole data run including the controllers.

The library

The left column lists what you can put on the canvas. **Drag** an item onto the canvas, or **click** it and then click where it should go.

Pixel Bar Pro · 24 V — straight bars in four lengths and a 90° corner piece. Pixel counts and power are in the library card and carry into the rig stats.

2D Connectors — the joints between bars: straight, 90°, 120°, T, 3-way triangle, 4-way cross, 6-way cross, and an adjustable elbow whose angle you type in the inspector. Connectors carry no pixels; they are geometry and, on the pull list, hardware.

Controllers live in the Wire tab's library, not here, because they are part of the data run rather than the picture.

The library. Bars first, then connectors.

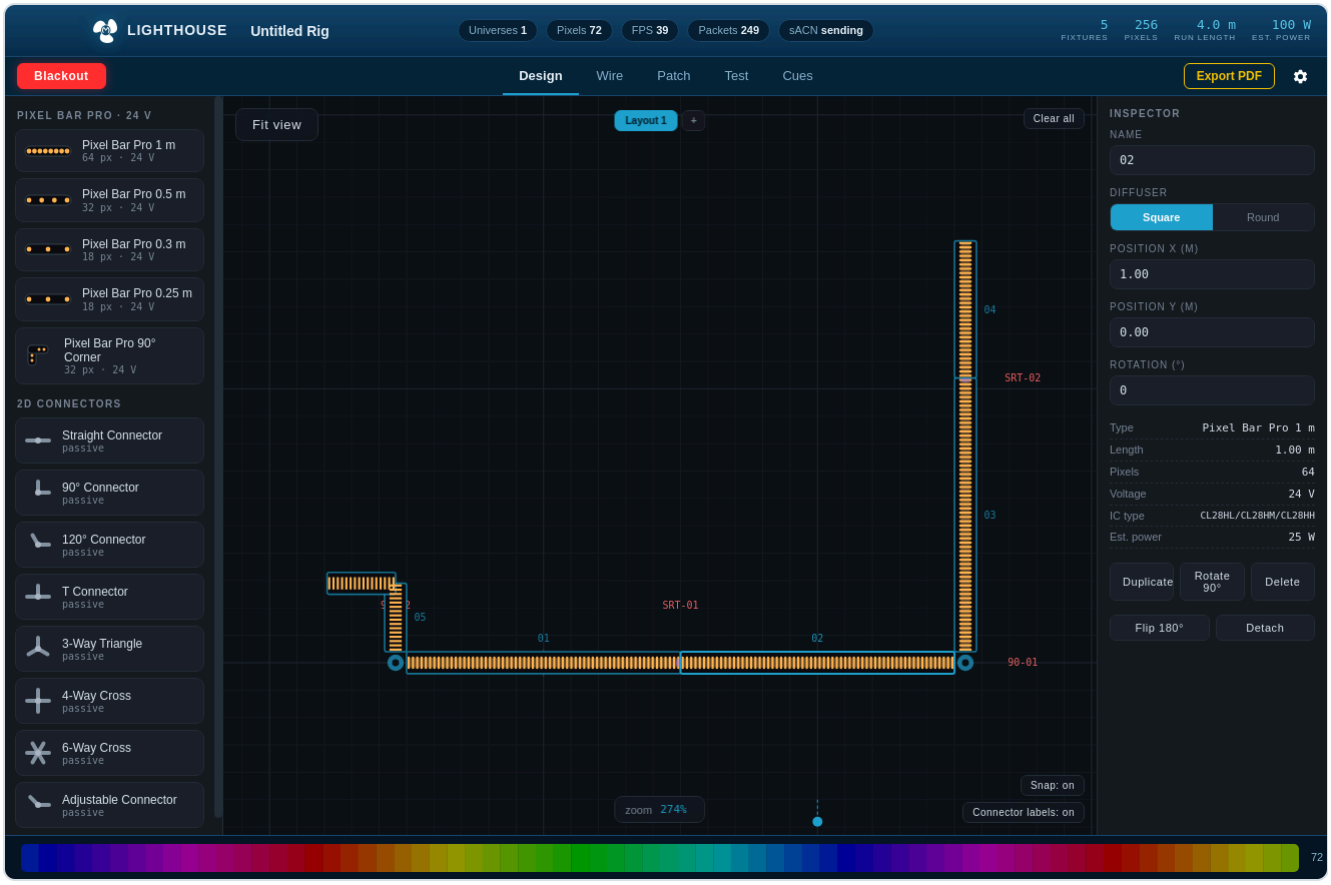
Placing and joining

Bars and connectors are magnetic to each other within about 20 cm. Drag a bar's end toward an open socket on a connector and it seats itself and rotates to match the arm. Drag a connector toward the free end of a bar and it seats on that end, turned so one arm points back into the bar. You do not have to line anything up by hand.

- **Straight connector** joins two bars end to end with no gap. Every other connector is an elbow with a 4 cm hub, so bars on either side sit 4 cm off its centre, like the real part.
- **Corner bars** seat into connectors the same way — either leg can plug in. Rotate the corner before you drop it to choose which way the L turns.
- A connector that already joins two bars is a **protected joint**: nothing else snaps onto it and it cannot be swapped out. Delete one of the bars first if you need to change it.
- Dropping a connector onto a lone connector **replaces** it — handy for swapping a T for a cross without disturbing the bars.

Solid pieces never overlap. If a drop would put a bar through another bar, it lands beside it instead.

Selecting and editing



Bar 02 selected. The inspector on the right edits the selection; the whole assembly it belongs to is outlined so you can see what will move together.

Click a piece to select it. Click an empty spot to deselect. Drag a selected piece to move it — and note that pieces joined by connectors move as one rigid assembly, so dragging bar 02 above brings 01, 03, the connectors and the corner with it. Hold ⇧ and click to build a selection of separate pieces; they move together but do not snap.

The **Inspector** shows the selection's name, diffuser (square or round), position in metres, and rotation. Below that are the catalogue facts for the part — type, length, pixels, voltage, LED IC, estimated power — and buttons to **Duplicate**, **Rotate 90°**, and **Delete**. For bars there is also **Flip 180°**, which swaps the bar's data-in and data-out ends without moving it; you will care about this in Wire.

Names are automatic: bars count up `01, 02, 03…` across the whole rig; connectors and controllers count per type (`SRT-01`, `90-01`, `CL404R-01`). Rename anything from the inspector.

Layout pages

The tabs at the top centre of the canvas are layout pages. **+** adds one. A rig can carry several — the lounge and the main stage, or three options for the same wall — and each page has its own drawing, wiring and patch.

Double-click a page tab to rename it; the × on it deletes it.

> **CLEAR ALL**
> Removes every fixture and cable from the current page. Because that is a lot to lose to a stray click, the dialog makes you type `yes`. It is still undoable with `⌘Z` straight afterwards.

CHAPTER 4

Wire — cable it to a controller

Wire is where the drawing becomes a patch. Put a controller on the canvas, cable its ports to the bars in the order the data actually runs, and Lighthouse addresses every pixel for you.

The demo rig wired. Port 1 of the CL-404R feeds bars 01 → 02 → 03 → 04 in a chain; port 2 feeds the corner on its own. Each port has a colour, and every bar on that port takes it.

What changes on this tab

Layout is locked — you cannot move or rotate bars here, only wire them. Controllers appear, connectors fade back, and each bar shows its two data terminals: **M** for the male input and **F** for the female output. The library on the left now lists controllers.

> **UNWIRED BARS ARE NOT PATCHED**
> A bar with no cable into it has no universe and no address, so it is left out of the patch and stays dark in Live view. This is deliberate: Lighthouse never invents an address for something you have not told it about. If a bar is dark that should not be, come back here and look for the missing cable.

Controllers

Drag a **CL-404R Controller** in from the library (or the generic 8-port controller if you are driving something else). Its output ports hang off the bottom edge, numbered and colour-coded. A controller describes its own addressing — how many ports, how many pixels and universes per port — so the patch it produces matches what you configure on the box.

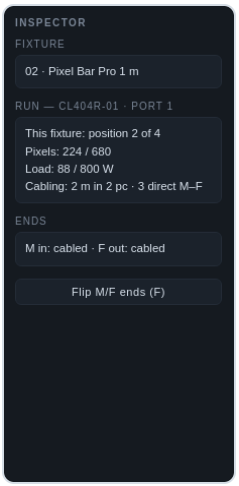
Select a controller to set its **base universe** (the first universe on port 1) and **pixel grouping** (how many physical LEDs act as one control pixel, if the controller is set up that way). Leave the base blank and Lighthouse assigns one automatically.

Running cables

1. Drag from a controller port and drop on a bar's **M** terminal. That bar is now first on that port.
2. Drag from that bar's **F** terminal to the next bar's **M**. Keep going down the run.
3. Dropping on a terminal that already has a cable replaces it. Click a cable and press ⌫ to remove it.

Cables only connect F to M, the same as the hardware. If a bar's ends are backwards for the direction the data needs to go, click the bar and press F (or use **Flip M/F ends** in the inspector) — the bar stays put and its terminals swap.

The label on each cable is its length. It is measured on the drawing, terminal to terminal, so a cable that runs a long way round in the real world will read short here; the number is for sanity, not for ordering.



A bar's inspector on Wire. Which controller and port it is on, its position in the chain, and how much of that port's pixel budget the chain has used.

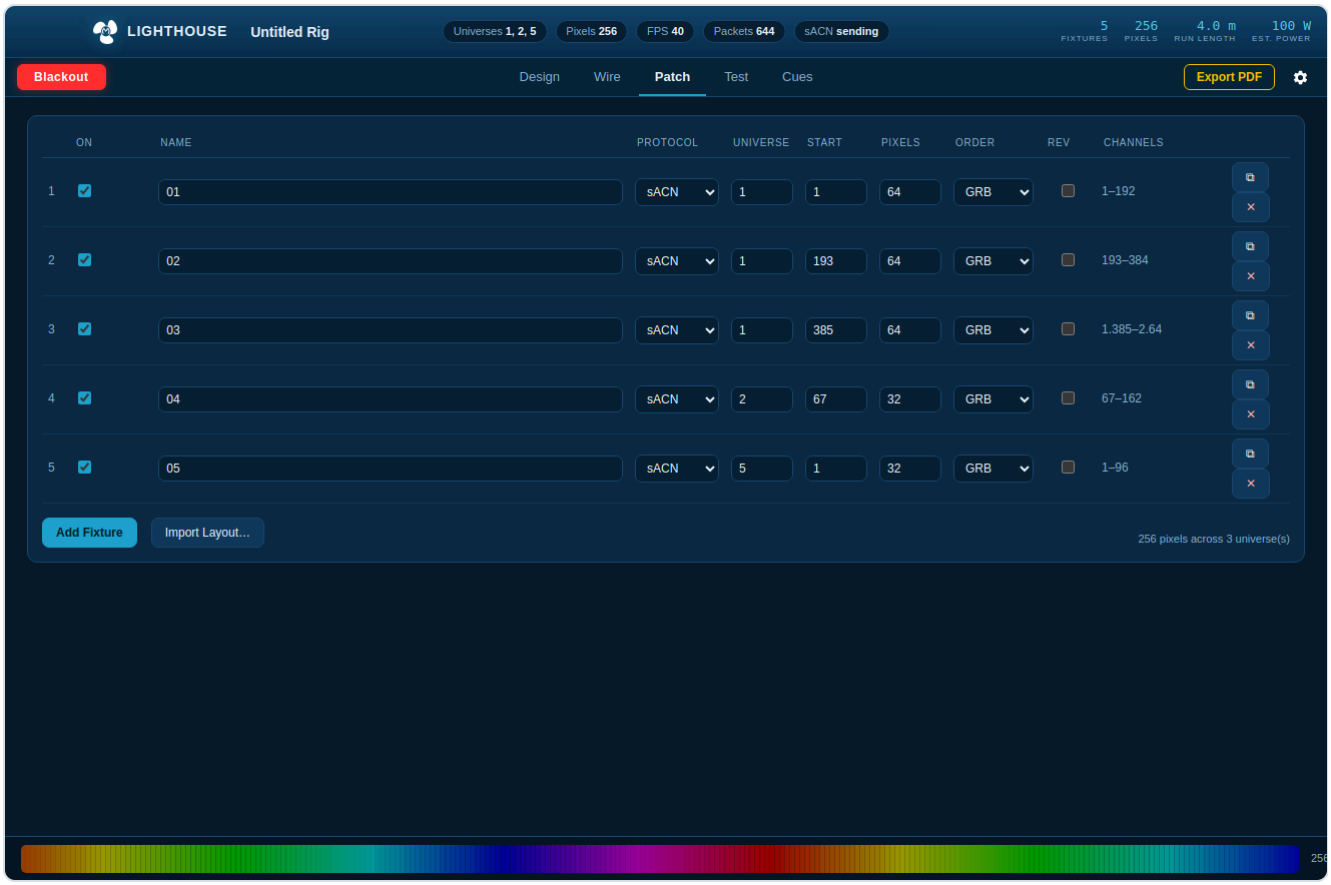
Port budgets

Every port has a pixel capacity and a power capacity (680 pixels and 800 W on the CL-404R). As you cable bars onto a port, the inspector for any bar in that chain shows the running total. Go over and the chain turns red and the Patch tab lists a warning. Lighthouse will not stop you — sometimes the plan is to fix it at the controller — but it will not let you miss it.

CHAPTER 5

Patch — what goes where

Patch is the table the engine renders into. When you wire the rig, Lighthouse fills this in. You can also type a patch by hand for a rig you have not drawn.



The patch from the demo rig. Bars 01–04 run continuously from universe 1 channel 1 through universe 2; the corner on port 2 starts at universe 5.

Reading the table

COLUMN	MEANING
On	Untick to take a fixture out of the render without deleting it.
Name	Matches the label on the canvas.
Protocol	sACN or Art-Net, per fixture. Half a rig can be on each.
Universe	1-based, sACN style. Art-Net fixtures are converted on the way out (see Settings).
Start	First DMX channel of the first pixel.
Pixels	Control pixels in the fixture.
Order	Colour order the fixture expects. CLEN bars are GRB.
Rev	Reverse pixel direction for this fixture only.
Channels	The span this fixture occupies, so you can see at a glance where the next one starts.

Overlapping addresses and anything that runs past channel 512 are flagged in red. A bar that spans a universe boundary is fine — pixels pack continuously and the engine splits the packets — and the Channels column shows both universes.

How the automatic patch is built

Each port of a controller is a block of universes (four on the CL-404R). Port 1 of the first controller starts at universe 1, port 2 at 5, port 3 at 9, port 4 at 13. Within a port, bars are addressed in cable order, 3 channels per pixel, 170 pixels per universe, running straight on from one bar to the next. A second controller starts at universe 17.

That means a patch with only a few bars will show gaps — universes 1 and 5 with nothing between them. That is intentional: the numbers line up with the controller's own port windows, so the universe you see here is the universe you type into the controller's setup page. To start a controller somewhere else, set its base universe in Wire.

Patching by hand

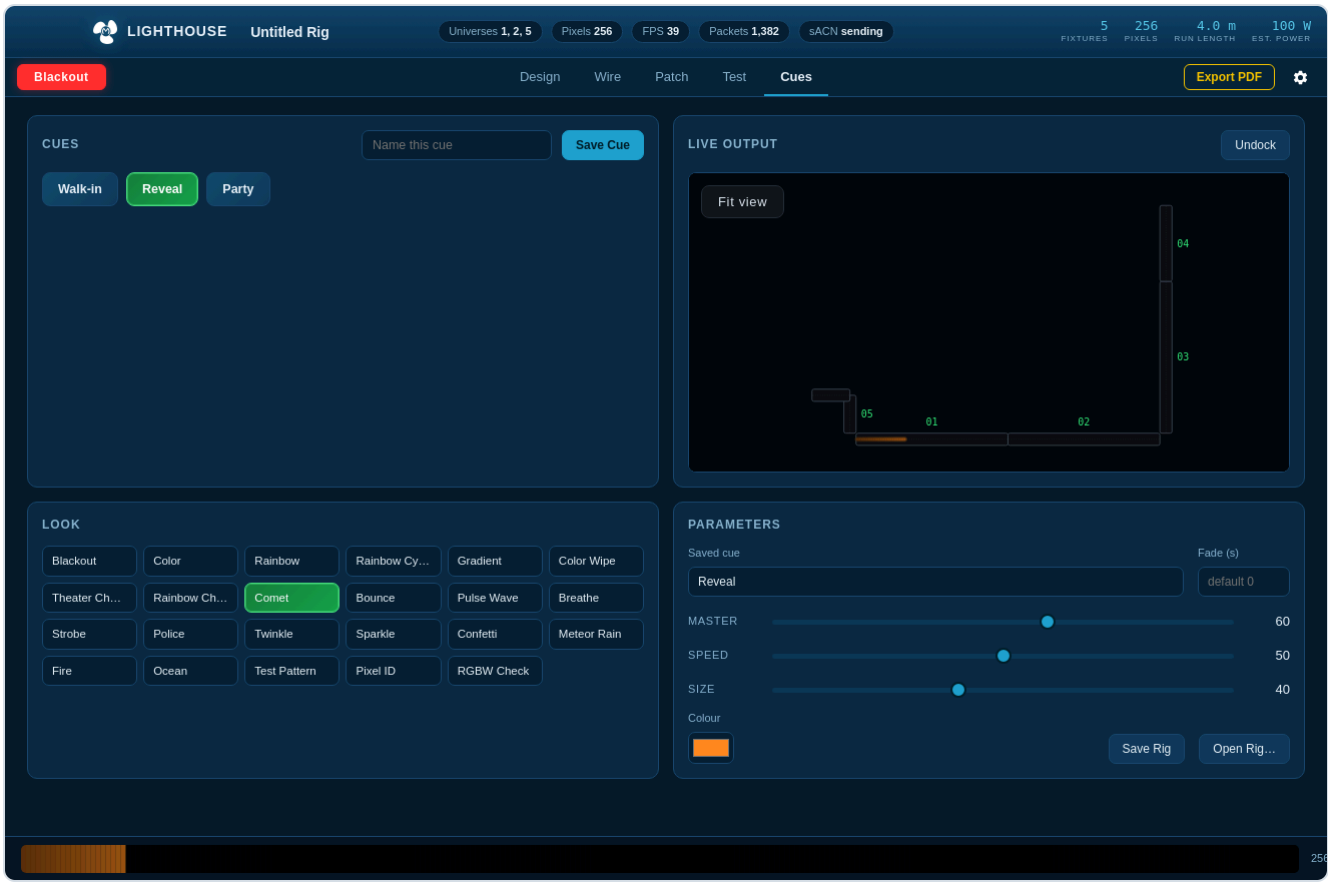
Add Fixture appends a row you fill in yourself. **Import Layout…** reads a JSON fixture list from another tool. Both are for rigs you are not drawing; if the rig is drawn and wired, the table is regenerated from the drawing whenever the wiring changes, and hand edits to wired fixtures will be overwritten.

An empty drawing never wipes a hand-built patch. If you have typed a patch and then start drawing, the patch is replaced only once something on the canvas is actually cabled to a controller.

CHAPTER 6

Cues — run the show

Cues is the page you leave open during the event. Pick a look, shape it with the sliders, save it as a named cue, and watch the rig respond in Live output.



Cues. Three saved cues top left — *Reveal* is on stage, so it is green — the live monitor top right, the 23 built-in looks bottom left, and the parameters bottom right. Comet is running.

Looks

The **Look** panel is the 23 built-in effects. Click one and it is live immediately — the one on stage is green. They render across the whole rig as one continuous strip in patch order, so a comet crosses every bar in cable sequence instead of restarting on each.

Blackout — everything off. **Color** — one flat colour. **Rainbow** — the full spectrum spread along the rig. **Rainbow Cycle** — spectrum scrolling along the run. **Gradient** — two-colour blend. **Color Wipe** — colour fills the run then clears. **Theater Chase** — marquee dots. **Rainbow Chase** — the same, in spectrum. **Comet** — a head with a fading tail. **Bounce** — a bar of light ricocheting end to end. **Pulse Wave** — waves of brightness travelling the run.

Breathe — slow fade up and down. **Strobe** — hard flashes. **Police** — red / blue alternating halves. **Twinkle** — random pixels glowing and fading. **Sparkle** — sharp white pops over a colour. **Confetti** — random colours popping in. **Meteor Rain** — falling streaks with decay. **Fire** — flicker in reds and oranges. **Ocean** — slow blues and teals. **Test Pattern** — every patched fixture in its own colour, first pixel white, so the programmer can see which bar is which from the floor. **Pixel ID** — every pixel a different colour, for finding a dead or repeated pixel. **RGBW Check** — red, green, blue and white blocks for checking colour order.

Parameters

Parameters. The saved cue's name and fade on top, then master, speed, size and colour.

Master scales the brightness of everything. **Speed** sets how fast the look moves and is remembered per look, so a slow Ocean and a fast Strobe do not share a slider. Some looks add **Size** (how long a comet's tail is, how wide a chase dot is) or a colour picker. The sliders follow your hand — drag and the rig moves with you; there is no apply button. Nudging a slider does not un-select the saved cue: it is still that cue, adjusted, until you pick another.

The row above the sliders belongs to the saved cue on stage. **Saved cue** is its name — type a new one and press Enter to rename it. **Fade (s)** is how many seconds the rig takes to crossfade into this cue when it is recalled; leave it blank to use the rig's default fade from Settings. Both fields grey out when you have picked a look directly rather than a saved cue.

Saved cues

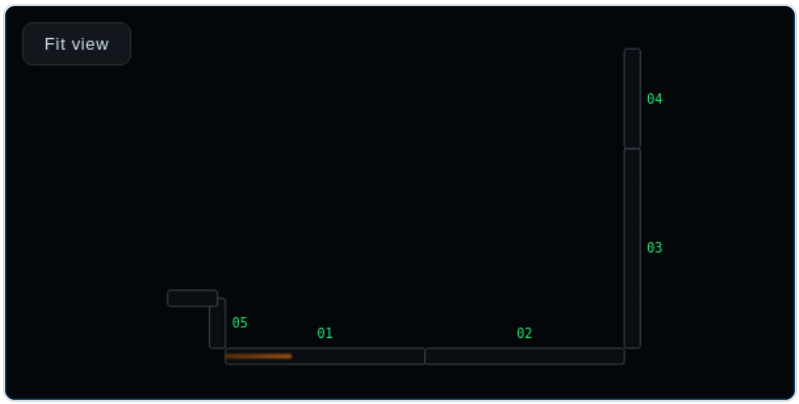
Saved cues. The green one is on stage.

When the look is right, type a name in **Name this cue** and click **Save Cue**. It becomes a button in the Cues panel that recalls the look, colour, speed and size together. Right-click a saved cue to rename it, set its fade, update it to the current settings, or delete it. Saved cues travel with the rig file; they are what the Control API and a Companion button call by name, and what a console recalls by number (chapter 11) — number 1 is the leftmost button.

Fades

A recall crossfades: the outgoing look keeps running and dissolves into the incoming one over the cue's fade time, pixel by pixel, so a Comet melting into Ocean looks like a lighting cue and not a cut. The order of precedence is the fade sent with the call (Control API or console), then the cue's own fade, then the rig's **Default fade** in Settings. Zero means a snap. Clicking a look directly also fades, using the default. **Blackout** never fades — it is a snap in both directions, because when you need it you need it now.

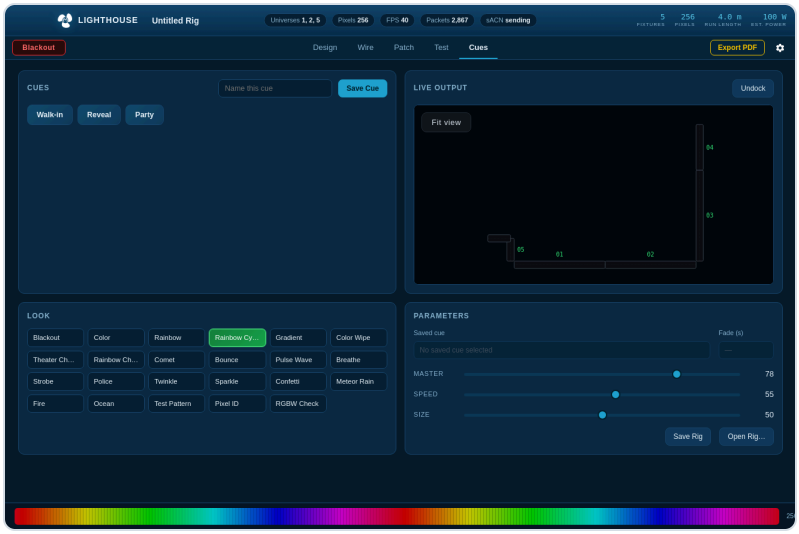
Live output



Live output paints every pixel from the DMX the engine is sending right now.

The monitor is your drawing with each pixel coloured from the actual frame going out the wire. It is a monitor, not a simulation: if a bar is dark here, the data going to it is dark. Connectors and controllers are hidden so you see the light and nothing else. **Fit view** reframes it; **Undock** (or `⌘L`) opens it in its own always-on-top window you can park on a second screen.

Blackout



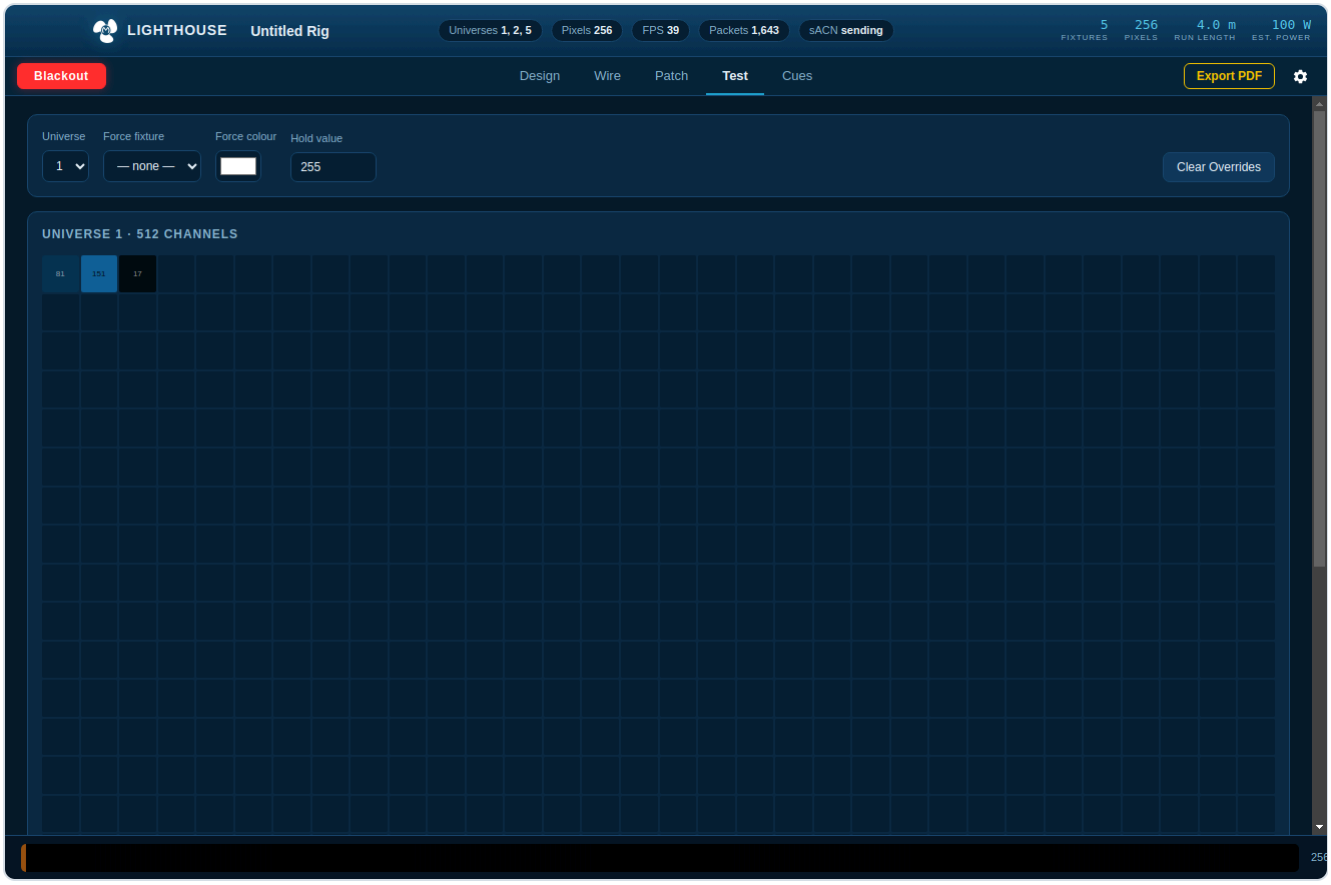
Blackout engaged. Output is zero, Live is dark, and the button flashes until you release it.

The red **Blackout** button and `⌘B` work from every page. Blackout does not change the current look or any slider; releasing it brings back exactly what was running.

CHAPTER 7

Test — prove the wiring

Test shows the numbers actually being written to a universe, channel by channel, and gives you ways to push known values at a fixture. It is how you find out whether a problem is in the Mac or downstream of it.



Universe 1, all 512 channels, live. Values update every frame. The fixture map underneath says which channels belong to which bar.

The channel grid

Pick a universe at the top left. Each cell is one DMX channel with its current value; the cells are shaded by level so a pattern is visible at a glance. Because this is the frame the engine is sending, if the channels here are moving and the fixtures are dark, the Mac is doing its job — look at the network, the controller's universe setting, or the cabling.

Holding channels

Click a channel to pin it at the value in the **Held value** box (255 by default). It stays there through any look until you release it. Pinned channels are outlined in orange. **Right-click** a pinned channel to release it, or use **Clear Overrides** to release everything at once.

Forcing a fixture

Force fixture picks a fixture from the patch and **Force colour** paints all of it one flat colour, overriding the running look for that fixture only. This is the quickest way to answer "is bar 03 really bar 03?" — force it red and look at the truss. Choose — *none* — to stop forcing.

Fixture map

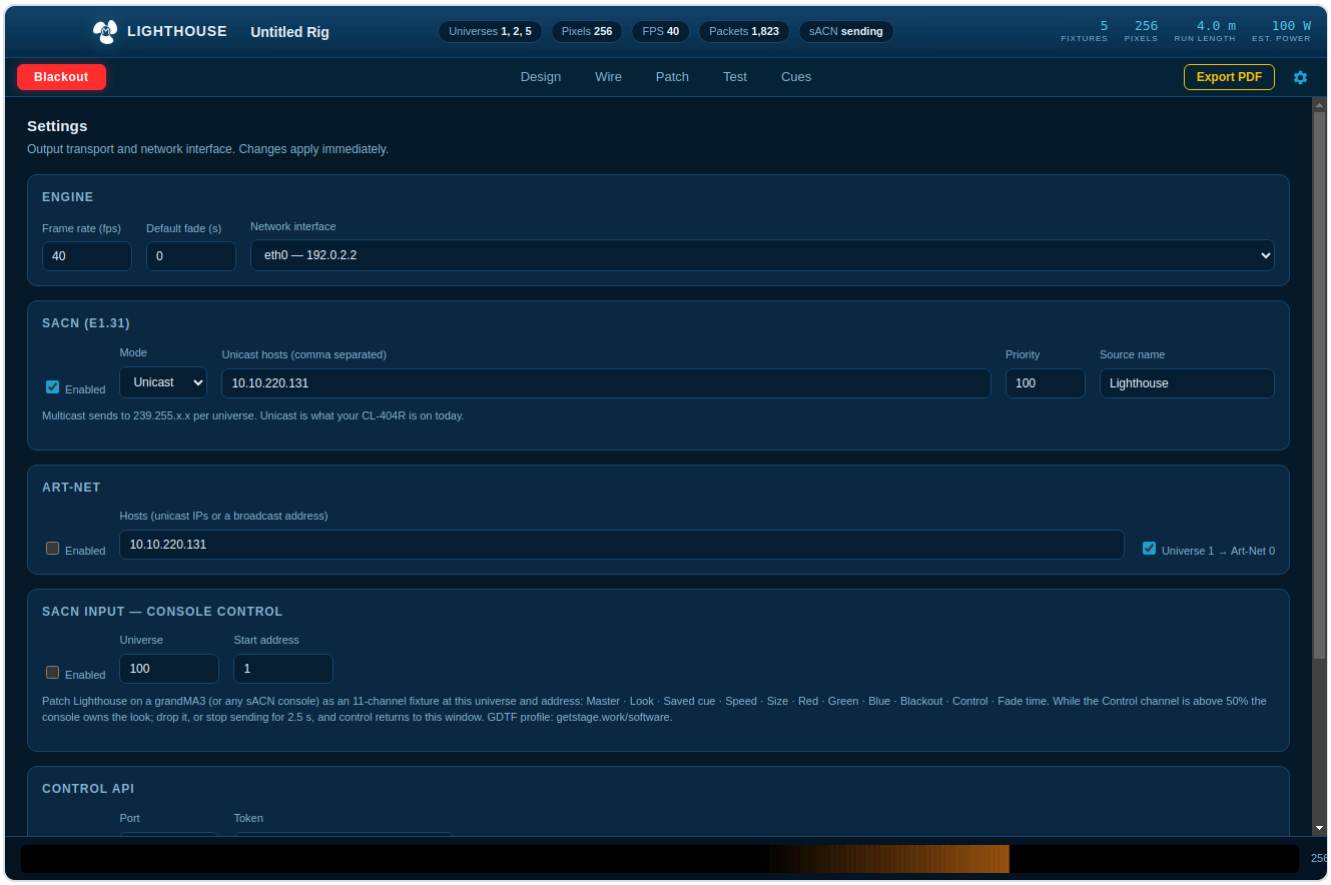
The table lists every fixture on the selected universe with its channel span, pixel count and colour order — the same facts as the Patch tab, filtered to what you are looking at.

> **CHECKING COLOUR ORDER**
> Run the **Test Pattern** look from Cues. It sends red, green, blue, white blocks in that order. If a bar shows green first, its order is GRB and the patch says RGB (or the other way round). Fix it in the Order column on Patch.

CHAPTER 8

Settings — network and output

Settings is where Lighthouse meets the controller. Every change here applies immediately — there is no save button — and a "Settings applied" toast confirms it.



Settings. Engine, sACN, Art-Net, sACN input, the Control API and the rig file.

Engine

Frame rate is how many frames per second the engine renders and sends. 40 is the default and right for pixel bars; 30 is fine if the Mac is struggling, 44 is the sACN maximum. **Default fade** is the crossfade, in seconds, used by any recall that does not bring its own — 0 snaps. **Network interface** is which NIC the packets leave through. *Auto* lets macOS choose. On a Mac with Wi-Fi and a wired show LAN, pick the wired one explicitly — multicast in particular leaves through exactly one interface, and macOS does not always pick the one you meant.

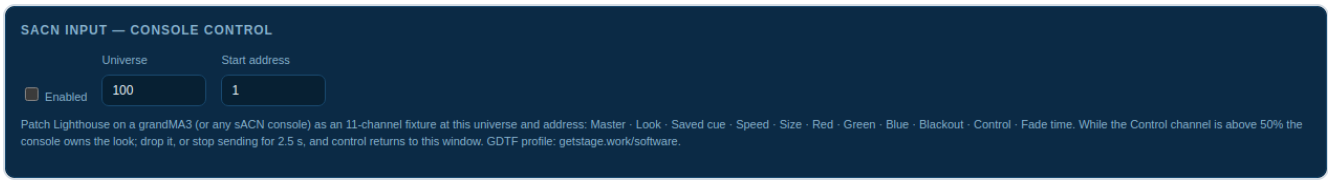
sACN (E1.31)

- **Enabled** — turn the protocol on or off. Fixtures set to sACN on the Patch tab only send while this is on.
- **Mode** — *Unicast* sends straight to the addresses in the host list. *Multicast* ignores the host list and sends each universe to its multicast group (239.255.x.x); any controller listening for that universe picks it up.
- **Unicast hosts** — comma-separated IPs of the controller(s). Ships set to `10.10.220.131`, the CL-404R's default. Change it, tab out, and the engine rebinds.
- **Priority** — sACN priority, 0–200, default 100. A controller receiving the same universe from two sources takes the higher priority; a backup laptop at 90 will take over automatically if the main one at 100 drops off.
- **Source name** — how this Mac identifies itself in the packets. Shows up in controller diagnostics.

Art-Net

- **Enabled** and **Hosts** as for sACN. A broadcast address such as `10.10.255.255` reaches everything on the subnet.
- **Universe 1 → Art-Net 0** — Art-Net numbers universes from 0 where sACN starts at 1. With this on (the default) the patch's universe 1 goes out as Art-Net port address 0, universe 17 as 0/1/0, and so on. Turn it off only if your Art-Net gear displays universes 1-based and you want the numbers to match on both screens.

sACN input — console control



sACN input. Which universe and address Lighthouse listens on as a fixture.

Turn this on and Lighthouse becomes an 11-channel fixture that any sACN console can patch and drive — a grandMA3, an ETC Eos, a Hog. **Universe** and **Start address** are where the console should patch it; 100 / 1 by default, deliberately far from the output universes. Chapter 11 has the channel map and the GDTF file.

Control API

Control API. Port and optional token.

An optional HTTP server on the port shown (8080 by default) so something else on the LAN — Bitfocus Companion, Home Assistant, a stream deck, a show controller — can call cues. Leave **Token** empty for open access on a network you trust, or set one and every client must send it. Chapter 10 has the calls.

Rig file

New Rig, **Open Rig…**, **Save Rig** and **Save Rig As…** — the same as the File menu. Chapter 9 explains what a rig file holds.

CHAPTER 9

Rig files and PDF export

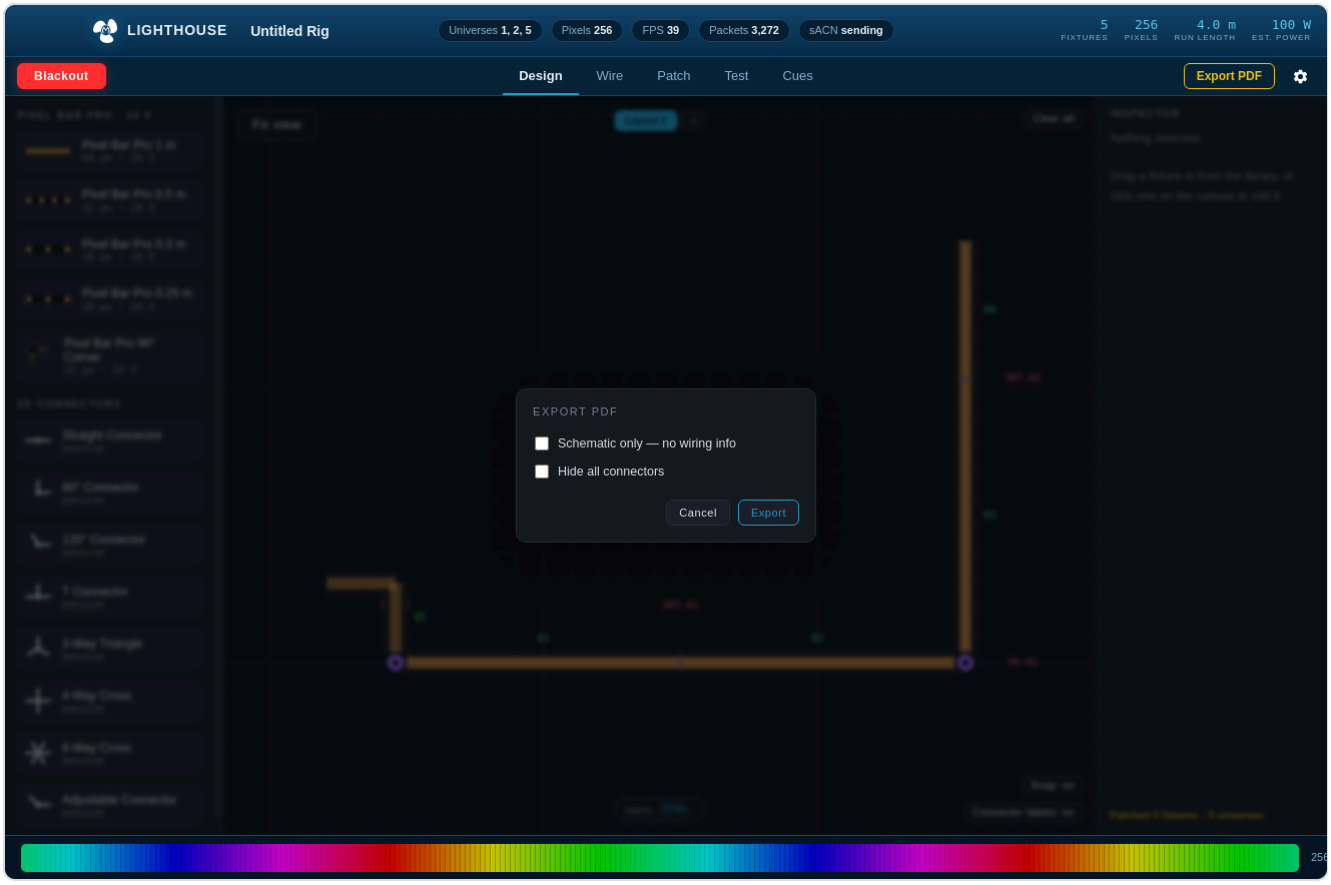
The rig file

A rig is one `.lighthouse` file: the drawing, the wiring, the patch, the saved cues, the parameters and the network settings. Hand it to whoever is running the show and they have everything.

Lighthouse autosaves the open rig continuously, so a crash or an accidental quit costs nothing. **Save Rig As…** (⇧⌘S) writes a named copy wherever you like; **Save Rig** (⌘S) updates it; **Open Rig…** (⌘O) loads one. The name in the title bar is the rig's name and becomes the title of any PDF you export.

Rig files are plain JSON. Older files from the first releases (patch only, no drawing) still open — they just have an empty canvas.

Export PDF



Export PDF. Two options, then the plot opens in your PDF viewer.

Export PDF in the tab strip prints the current layout page as a plot: the drawing to scale with every label, a pull list of bars and connectors by type and count, run length and power, and — if the rig is wired — a pixel patch showing every bar's controller, port, and sACN and Art-Net addresses.

- **Schematic only — no wiring info** leaves the controllers and patch pages out. Use it for the client or the scenic shop, who need the shape and not the addressing.
- **Hide all connectors** draws bars only, for a cleaner picture of the light.

CHAPTER 10

Control API

With the Control API enabled in Settings, anything on the network can drive Lighthouse with plain HTTP. Requests are JSON; every response is the current state. If you use Bitfocus Companion, you do not need any of this — the Companion module in chapter 11 wraps it.

CALL	BODY	DOES
`GET /api/status`	—	Everything: current look, parameters, blackout, the saved cue on stage, who is in control, whether a fade is running, output stats.
`GET /api/control`	—	The control state only — the same object the events stream sends.
`GET /api/events`	—	Server-Sent Events. One `control` event with the full state on connect, then one on every change, whoever made it. This is how Companion keeps its buttons lit.
`GET /api/looks`	—	The built-in looks with their ids.
`GET /api/presets`	—	The saved cues with their ids, in button order.
`POST /api/cue`	`{"id":"rainbow-cycle","fade":2}`	Switch to a built-in look. IDs are the look names in lower-case with hyphens: `color`, `theater-chase`, `pixel-id` … `fade` (seconds) is optional.
`POST /api/preset`	`{"name":"Walk-in"}` · `{"index":1}` · `{"id":"…"}`	Recall a saved cue by name, by 1-based button position, or by id. Optional `fade` overrides the cue's own.
`POST /api/params`	`{"brightness":80,"speed":40,"color":"#ff0000","size":50}`	Set any of master brightness, speed, colour, size. Omit what you are not changing.
`POST /api/blackout`	`{"on":true}` or `{"toggle":true}`	Engage, release or toggle blackout. Always a snap.

TOKEN

With a token set in Settings, every request must carry `Authorization: Bearer <token>`, or `?token=<token>` on the URL for clients that cannot set headers. Without one, the API trusts anyone who can reach the port — keep it on the show LAN, off guest Wi-Fi, and disabled when you do not need it.

Example — a stream deck button that recalls "Walk-in"

```
curl -X POST http://10.10.221.10:8080/api/preset \
     -H "Content-Type: application/json" \
     -H "Authorization: Bearer mysecret" \
     -d '{"name":"Walk-in","fade":3}'
```

Example — Home Assistant rest_command

```
rest_command:
  lighthouse_blackout:
    url: http://10.10.221.10:8080/api/blackout
    method: POST
    content_type: application/json
    payload: '{"on": true}'
```

CHAPTER 11

Remote control — console, Companion, GDTF

Lighthouse does not have to be run from its own window. A lighting console can own it as a fixture over sACN, a Stream Deck can fire cues through Bitfocus Companion, and both can share the stage with the Mac. Whoever changed the look last, the window shows it.

Remote. While a console holds Lighthouse, a yellow pill says so in the title bar.

From a console — sACN input

Enable **sACN input** in Settings, note the universe and address, and patch Lighthouse on the console as an 11-channel fixture there. The GDTF profile `Maritime@Lighthouse@Control_11ch.gdtf` from getstage.work/software does that in one import on a grandMA3; on a console without GDTF, patch a generic 11-channel fixture and use the map below.

CH	FUNCTION	VALUES
1	Master	0–255 → brightness 0–100 %.
2	Look	Bands of 10: 0–9 no change, 10–19 look 1 (Blackout), 20–29 look 2 (Color) … in the order of the Look panel. Acts when the band changes, so the console can leave it parked.
3	Saved cue	0 none, 1 = first saved-cue button, 2 = second … Acts when the value changes.
4	Speed	0–255 → 0–100.
5	Size	0–255 → 1–100.
6 · 7 · 8	Red · Green · Blue	The look's colour.
9	Blackout	≥ 128 engages, < 128 releases.
10	Control	≥ 128: the console owns the look. < 128: Lighthouse is back under local control.
11	Fade time	0–255 → 0–25.5 s, in tenths. Used by Look and Saved cue recalls.

Channel 10 is the important one. While it is high, the console's Master, Speed, Size and colour are applied every frame and win over whatever the cue stored; the window's sliders follow them. Drop it, or stop sending

for 2.5 s, and control returns to the window with the last look still running — nothing snaps to black when the console goes away. If two consoles send the same universe, the higher sACN priority wins, as it does for any fixture.

> **PROGRAMMING TIP**
>
> Store Control at full in every cue that touches Lighthouse and at zero in a "release" cue. Recall a saved cue by number on channel 3, then set Master and Fade — the saved cue brings the look, the console brings the levels.

From a Stream Deck — Bitfocus Companion

The Companion module `maritime-lighthouse` is on getstage.work/software as a module package. In Companion 4 or later: **Settings → Modules → Import module package**, choose the file, then add the connection with the Mac's IP, the Control API port and the token if you set one. The module talks to the Control API and subscribes to its event stream, so a button lit for "Walk-in" goes dark the moment the console takes over.

Actions: recall a saved cue by name or position, set a look, master and speed set or nudge, size, colour, blackout on/off/toggle, all with an optional fade. Feedbacks: blackout engaged, look active, saved cue active, remote in control, master above a level. Variables: the current cue, saved cue, master, speed, size, colour, blackout, fading, source and rig name. Presets are ready-made buttons for every saved cue and every look — drag them onto a page and you are done.

Who is in control

Every source goes through one control model, so there is no fighting. A console holding channel 10 owns the levels; the window and Companion can still fire cues, and the console's next frame re-asserts its levels. Blackout from anywhere is blackout everywhere. The events stream (chapter 10) reports `source` — `local`, `http`, `sacn` or `companion` — so anything watching knows who moved last.

CHAPTER 12

When nothing lights up

Work from the Mac outward. Each step tells you whether the problem is above or below it.

1. Is the engine sending?

Title bar: **Packets** should be climbing and the protocol pill should say **sending**. If it says **not sending**, the socket did not open — go to Settings and pick the network interface that actually reaches the controller, then tab out of the field. If Packets is frozen at 0 with a green pill, the patch is empty: nothing is wired (Wire) or every fixture is unticked (Patch).

2. Are the channels moving?

Test tab, the universe the fixture is on. Values changing means the Mac is rendering and writing that universe. If they are not, the fixture is not in the patch or is on a different universe than you think — check Patch.

3. Is it leaving through the right door?

Settings → Network interface. A Mac on Wi-Fi and a wired LAN will happily send everything out Wi-Fi. Choose the wired interface. If you are using multicast, also confirm the switch has IGMP snooping configured or disabled consistently — a switch that snoops with no querier will drop multicast after a few minutes.

4. Does the controller agree on the numbers?

The controller's own setup page: is its IP the one in the Unicast hosts field? Is it listening for the universe the Patch tab shows for that port? On Art-Net, remember the 0-based offset — Lighthouse universe 1 is Art-Net 0 unless you turned that off, and some controllers display Art-Net universes 1-based in their UI even though the wire is 0-based.

5. One bar dark, the rest fine?

Almost always wiring in the app, not the truss. On Wire, find the bar: is there a cable into its M end? Is it in the chain (same port colour as its neighbours)? A bar with no cable is unpatched and dark by design. If the cable will not attach, the bar's ends are backwards — select it and press `F`.

6. Colours wrong?

Run **Test Pattern** and compare the order on the bar to R-G-B-W. Fix the Order column on Patch. If only the last part of a chain is off, the controller's per-port pixel count is lower than the chain — everything past that count stays dark or repeats.

7. Flickering or strobing that goes away with a direct cable?

The controller's Ethernet port is 100 Mb and it will drop pixel frames under a flood of unrelated broadcast and multicast traffic — an office LAN with kiosks, printers and phones on it is enough. Put the Mac and the controller on their own VLAN (or their own switch) with nothing else on it, and use static addresses there. If

the Mac is on Wi-Fi for the show, stop: Wi-Fi jitter looks exactly like this too. Wired for shows, Wi-Fi for programming from the couch.

8. Local Network permission

If Lighthouse shows sending, channels move, the interface is right, and still nothing reaches the controller — and this is the first time this Mac has run it — check System Settings → Privacy & Security → Local Network and make sure Lighthouse is allowed.

CHAPTER 13

Reference

sACN and Art-Net in one paragraph each

sACN (E1.31) is the ANSI standard for DMX over Ethernet. Universes are numbered 1–63,999. It can unicast to a specific IP or multicast to a group per universe (239.255.hi.lo), which lets a managed switch deliver each universe only to the nodes that want it. Every packet carries a priority so a backup source can take over automatically. Lighthouse sends full E1.31 data packets with a stable source ID per rig.

Art-Net is the older de facto standard. Port addresses are 15-bit and 0-based, written Net/Sub-Net/Universe (0/0/0 is the first). It has no priority field. Lighthouse sends ArtDmx packets with 512 slots to the hosts you list, and converts from its 1-based universe numbers on the way out.

Fixture library

PART	LENGTH	PIXELS	NOTES
Pixel Bar Pro 1 m	100 cm	64	24 V, 25 W/m, GRB
Pixel Bar Pro 0.5 m	50 cm	32	
Pixel Bar Pro 0.3 m	30 cm	18	
Pixel Bar Pro 0.25 m	25 cm	18	
Pixel Bar Pro 90° Corner	2 × 25 cm	32	One L-shaped piece; either leg plugs into a connector
Straight Connector	—	0	Flush, no hub gap
90° / 120° / T / 3-way / 4-way / 6-way	—	0	4 cm hub
Adjustable Connector	—	0	Angle set in the inspector
CL-404R Controller	—	—	4 ports · 680 px and 800 W per port · 4 universes per port
Generic 8-port Controller	—	—	For other boxes; edit its numbers in the inspector

Addressing rules

- 3 channels per pixel; 170 pixels per universe (channels 1–510; 511 and 512 are unused).
- Pixels pack continuously along a port's chain and may cross a universe boundary mid-bar.

- Each port reserves a fixed block of universes (4 on the CL-404R), so port 2 starts 4 universes after port 1 whether port 1 filled its block or not.
- Controllers are numbered in label order; each starts one block after the last, unless you set a base universe.
- Universe numbers are sACN-style, 1-based. Art-Net output subtracts 1 by default.

Where things live on the Mac

WHAT	WHERE
Autosaved session	`~/Library/Application Support/Lighthouse/session.lighthouse`
Standalone layouts	`~/Library/Application Support/Lighthouse/layouts/` (deleted ones move to `layouts/deleted/`)
Downloads and this manual	getstage.work/software

Maritime Lighthouse is built by Maritime Creative, St. Augustine, Florida, for our own crews first. Found a bug or need something it does not do yet? Tell us at getstage.work. · Manual revision for Lighthouse 0.4.0, September 2026.