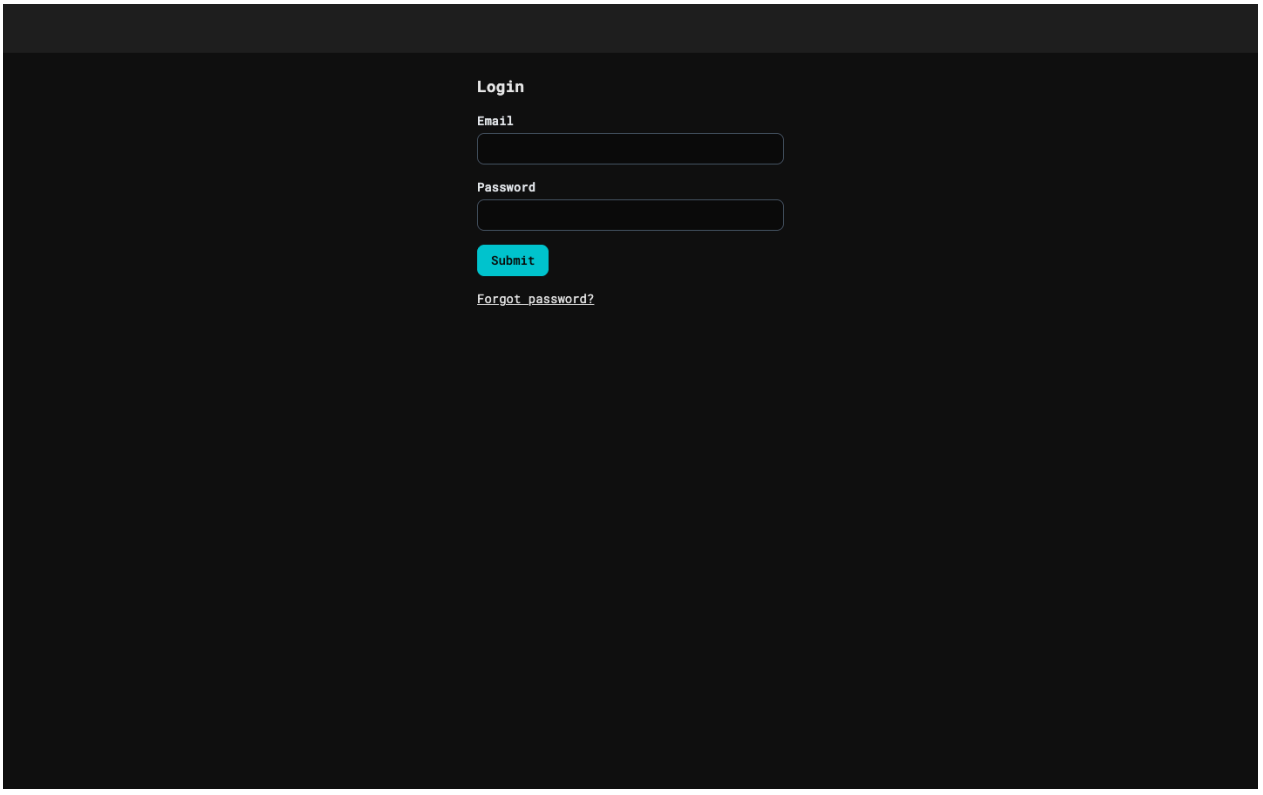
Radian Manual

Signing in

Open the editor at `localhost:5173` (not `127.0.0.1`). Without a session you land on the login page. Type your email and password and press Submit. Sessions last 30 days. Emails are not case-sensitive.



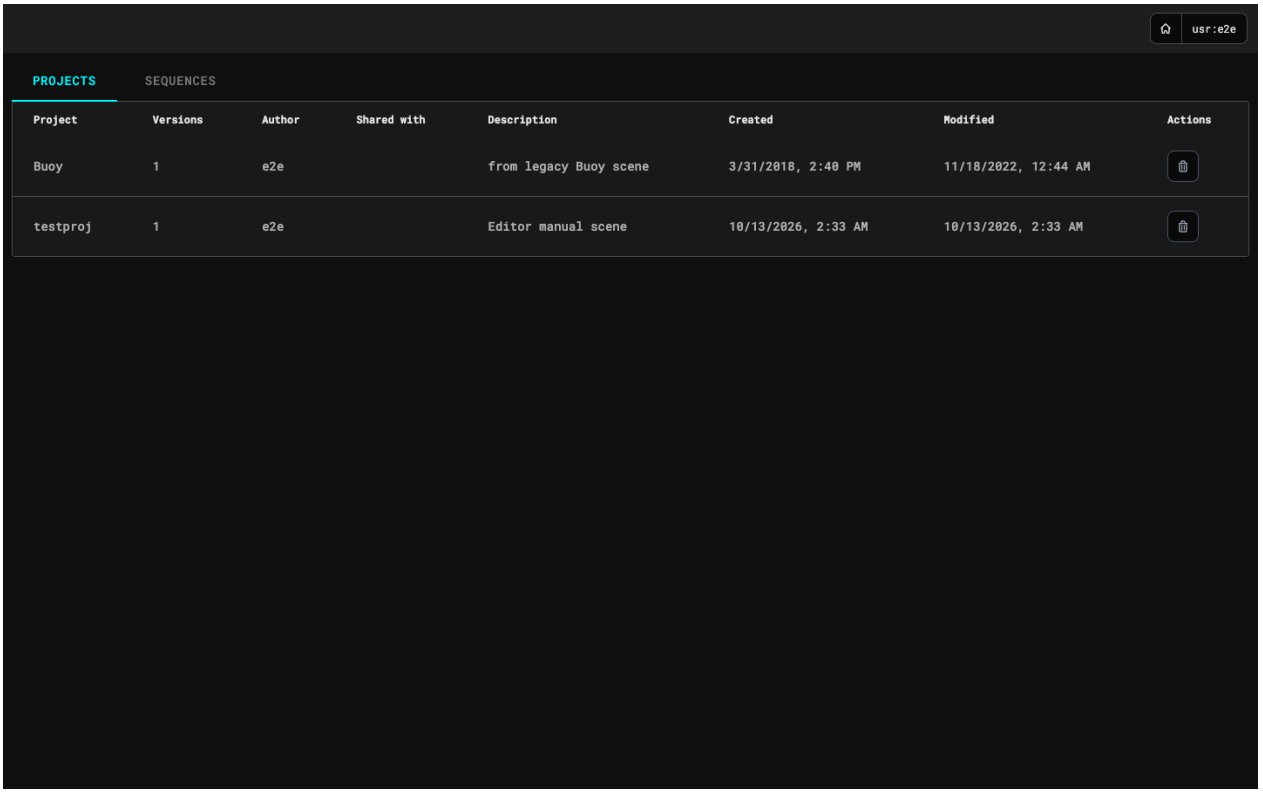
To sign out, open the `usr:<name>` button at the top right and pick Sign out.

The Profile page (the same menu) also has a Shader helper section for your Anthropic key: paste a key that starts with `sk-ant-` and press Save key. The key is stored encrypted and is never shown again; the page says "Key saved, ending" and its last four characters, and "This month" adds up what your helper calls cost. Remove key deletes it. Make the key in the Claude Console, in a workspace of your own with a spend limit.

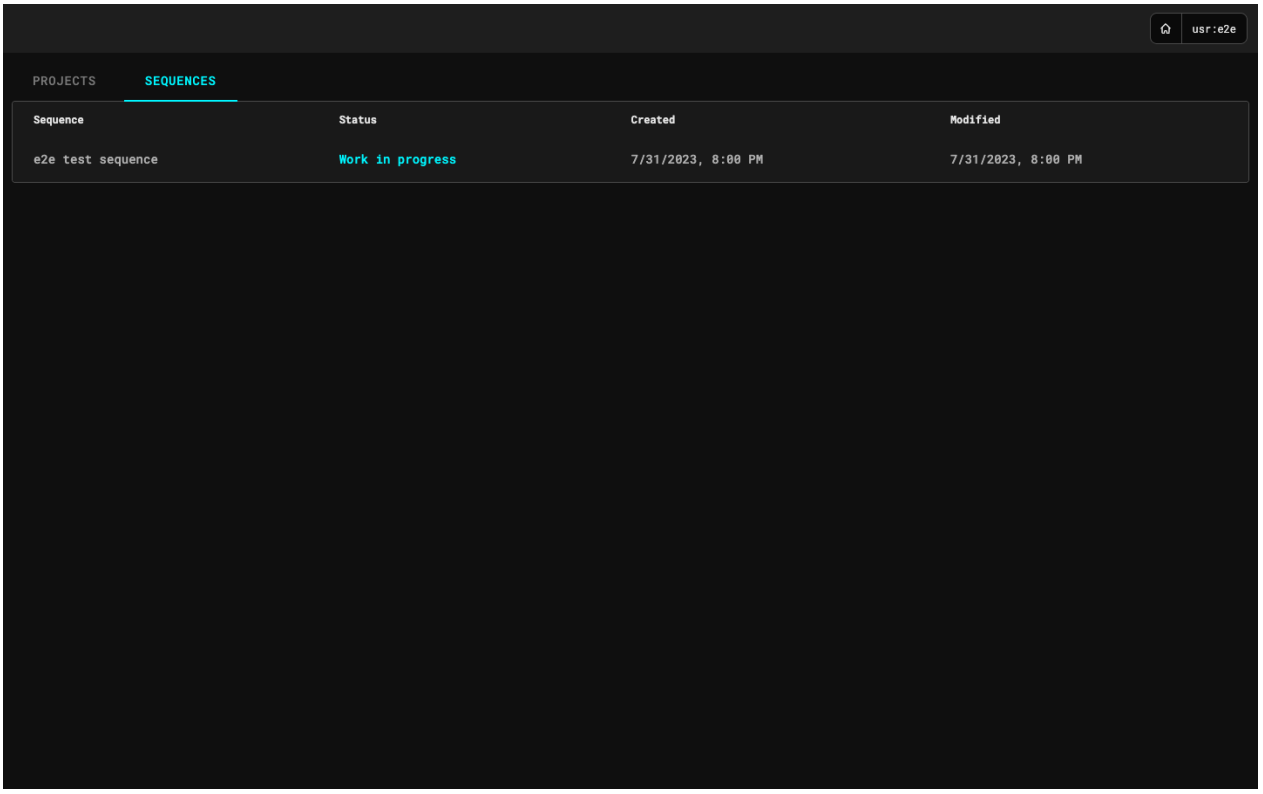
Below the key, Shader analysis chooses how the helper's suggestions for a shader's uniforms are asked for: Manual (the default) adds a button to the shader's Uniforms section, and Auto asks by itself when you open the Uniforms section of a shader material or replace its shader text, at most once for each shader text until the page is reloaded. Every ask is a call that your key pays for and is added to "This month". Your choice is saved as soon as you pick it.

Projects

The home page lists your projects. A project is a scene with its versions; the table shows the number of versions, the author, who it is shared with, the description of the newest version, and when it was created and last changed. Click a project's name to open it in the scene editor. The trash button at the end of a row deletes the whole project after you confirm.



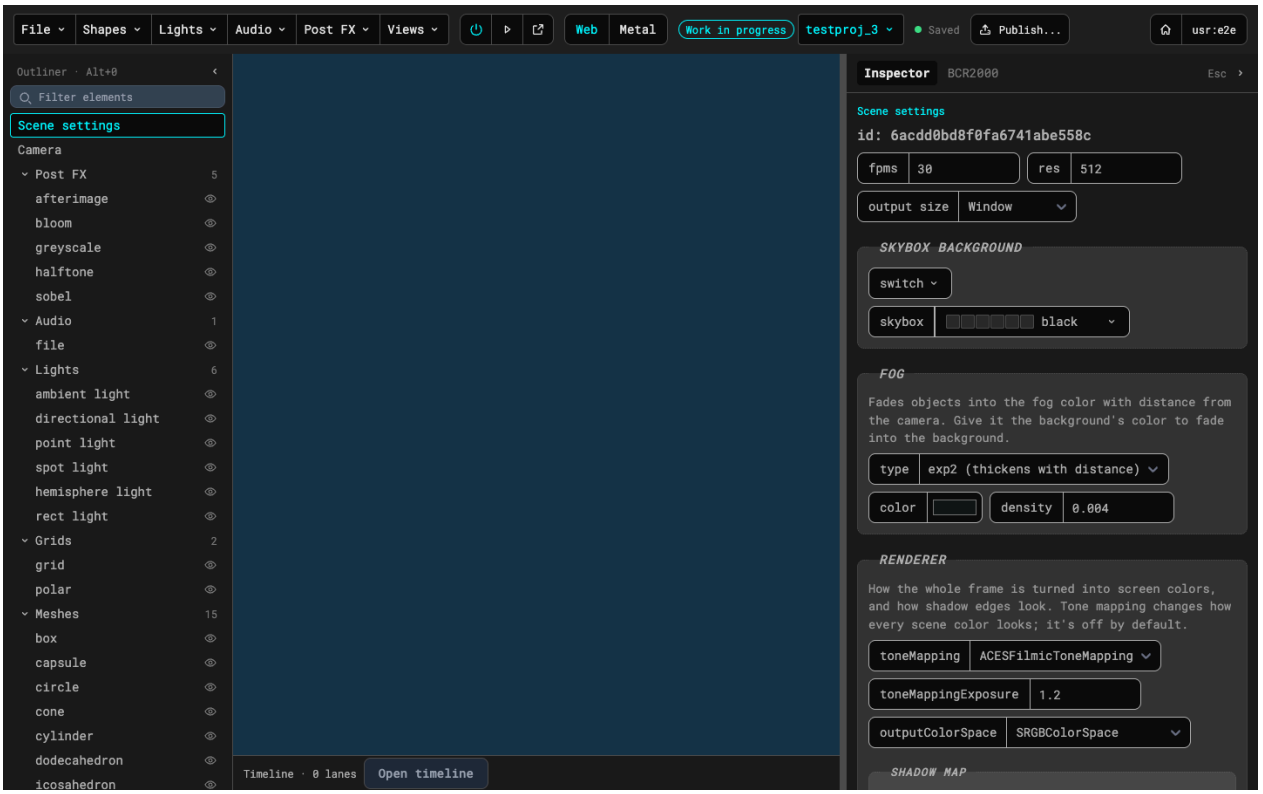
The Sequences tab lists sequences with their status and dates. Click a sequence to open it; see Sequences.



The home button at the top right (the house icon) brings you back here from anywhere.

The scene editor

Opening a project shows its newest version, unless the link names another one, in three parts under the toolbar. The address bar always names the open version (`?v=` and its id) and follows the Version menu, so a reload stays on the version you were editing and a copied link opens that version.



- **Toolbar** (top): the menus described in Toolbar menus, the preview buttons, the version's status and version menu, the save state, Publish, and your account.
- **Outliner** (left): the scene as a list. Scene settings and Camera come first, then one group for each kind of element (Post FX, Audio, Lights, Grids, Meshes, OBJs, Instance Geometries) with its count. Click a row to select it. Click a group's chevron to close or open it. Type in Filter elements to show only matching rows. The arrow keys move through the rows, Enter opens a group, Delete asks before it removes the selected element, and Alt+Up or Alt+Down moves the selected Post FX effect up or down one place. You can also drag a Post FX row onto another to move it.
- **Preview** (middle): where the scene plays once you start the preview, see Preview.
- **Inspector** (right): the settings of the selected row. The line at its top shows where you are, for example Meshes > box 1, and the element's key.

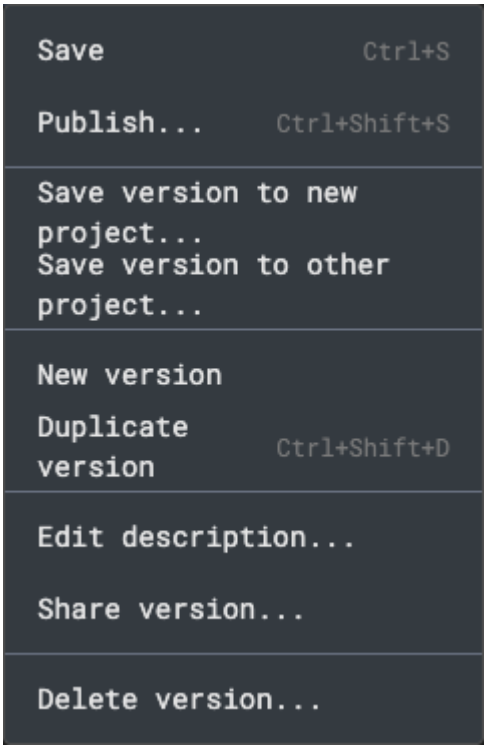
Each side panel has a chevron in its header that collapses it to a narrow rail; click the rail to bring it back. Drag the inspector's left edge to make it wider or narrower (320 to 720 px); with the edge focused, the left and right arrow keys do the same, and Shift makes the steps bigger. The editor remembers which panels are collapsed and how wide the inspector is, for each user.

Toolbar menus

Point at a menu or click it to open it. Click its name again to close it. The keys next to an item are its shortcut (see Keyboard shortcuts).

File

Save, Publish..., Save version to new project..., Save version to other project..., New version, Duplicate version, Edit description..., Share version... and Delete version.... What each does is in Saving, versions and status and Publish.



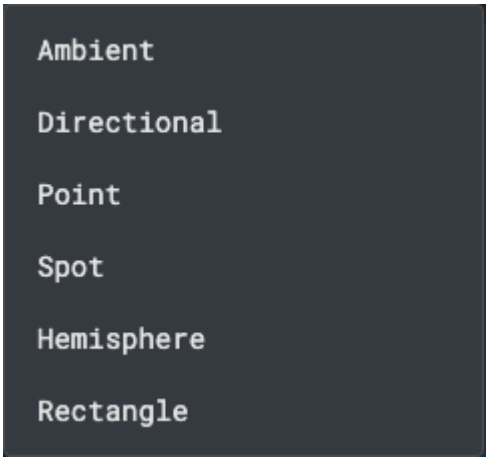
Shapes

Adds an element to the scene and selects it. Insert shape: Box, Capsule, Circle, Cone, Cylinder, Dodecahedron, Icosahedron, Octahedron, Parametric, Plane, Ring, Sphere, Tetrahedron, Torus and Torus Knot. Insert: OBJ file and Instance mesh. Insert grid: Grid and Polar grid.



Lights

Ambient, Directional, Point, Spot, Hemisphere and Rectangle.



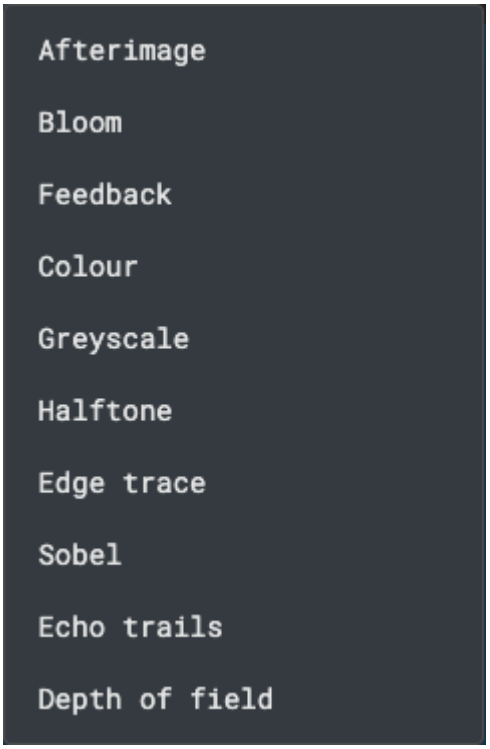
Audio

Adds a scene-level sound file. Oscillators are positional audio: add them from a mesh's or OBJ's insert menu.



Post FX

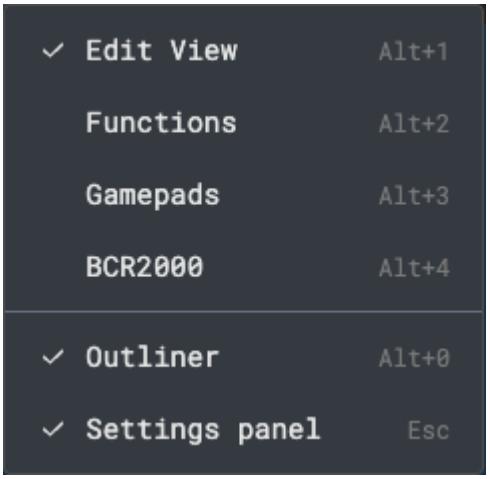
Adds a post-processing effect: Afterimage, Bloom, Colour (per-channel multiply, invert, wrap or fold and offset, then hue, saturation and lightness), Depth of field (f-stop and focal length, the largest blur, bokeh blades and angle, then the camera's Focus distance, Units per metre and an element for the focus to follow), Edge trace (vz.tracr's outline: edge contrast, line width and invert), Feedback, Greyscale, Halftone or Sobel. Depth of field blurs by distance from the camera's focus with a real lens model: Units per metre says how big a scene unit is (10 by default, so a 300-unit focus is 30 m away), and a new pass starts with Focus distance at 10, which makes most scenes soft, so set it to the distance of your subject. Focus on element makes the focus follow that element, so its own motion pulls focus. A dial or track on Focus distance has no effect while Focus on element is set, because the element's distance replaces it. Only the first Depth of field pass in a scene runs.



Views

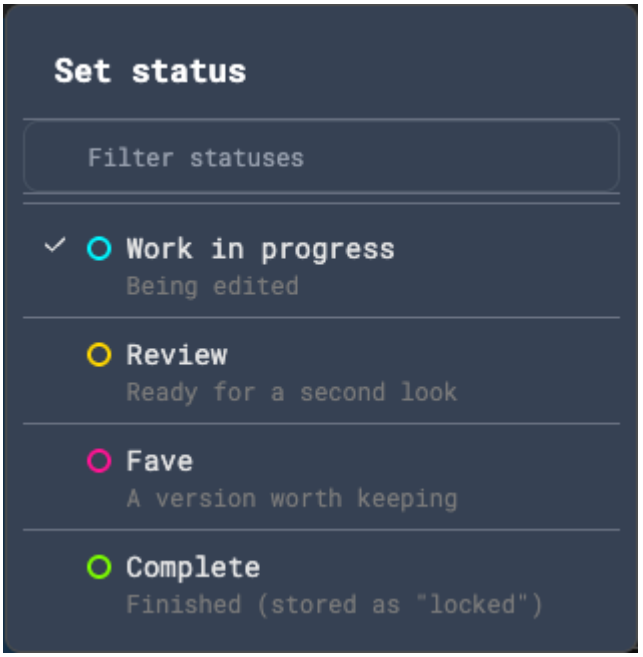
Switches the page: Edit View (the scene editor), Functions, Gamepads and BCR2000. The same menu shows or hides the Outliner and the Settings panel. A

check mark shows what is on.



Status

The coloured pill next to the version button shows the version's status. Click it to set a new one: Work in progress, Review, Fave or Complete. Type in Filter statuses to narrow the list. A status saves as soon as you pick it.



Version menu

The button next to the status, named for the open version (for example `testproj_3`), lists every version of the project with its description. Pick one to open it.

Account

The house button goes to the Projects page. The `usr:<name>` button opens a menu with Projects, Keyboard shortcuts and Sign out.

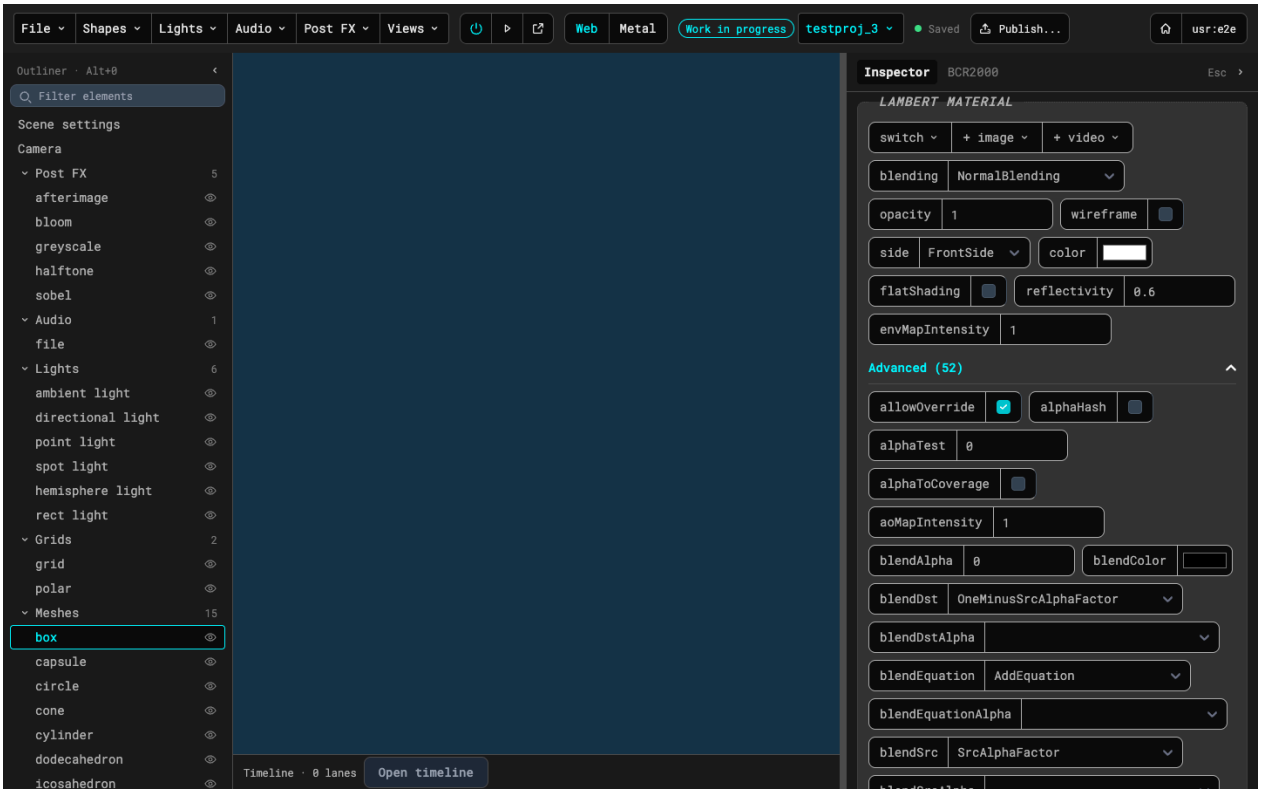
Editing elements

Select a row in the outliner to edit it in the inspector. Changes apply to the open version straight away. They are saved to the database only when you save; the toolbar says Unsaved until you do.

Settings rows

Every setting is a row: the name on the left and its field on the right. Number fields take typed values. Menus (such as blending) open a list. Rows that hold several numbers, like `pos x.y.z`, put one field per number. Colour rows open a colour picker. Point at a name to read what it does.

Settings used less often are folded into an Advanced row at the bottom of a panel. It shows how many settings are inside and how many are changed from their defaults, for example Advanced (6, 6 set). Click it to open it.

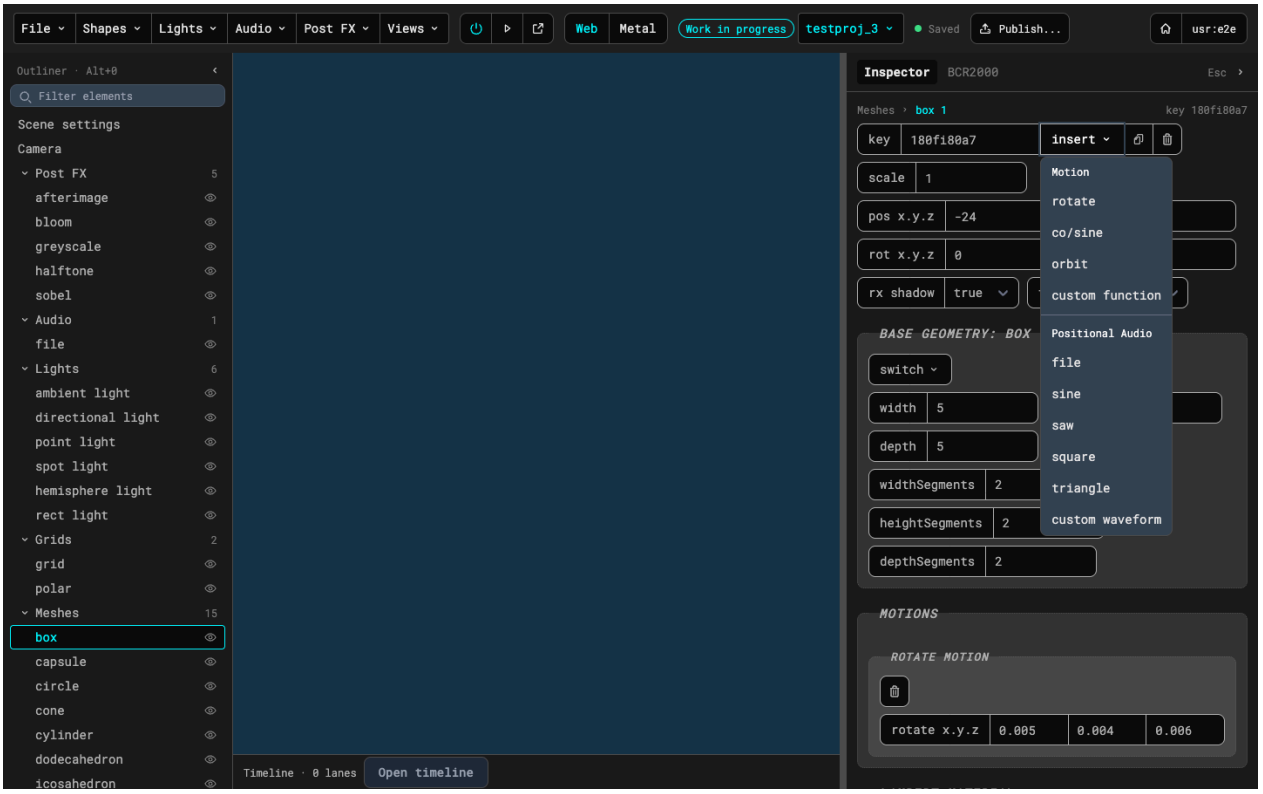


Element actions

Every element's panel starts with a row of actions: its key (the name the runner and Functions use to find it), an insert menu where the element can take one, a duplicate button that copies the element, and a remove button that deletes it.

The insert menu

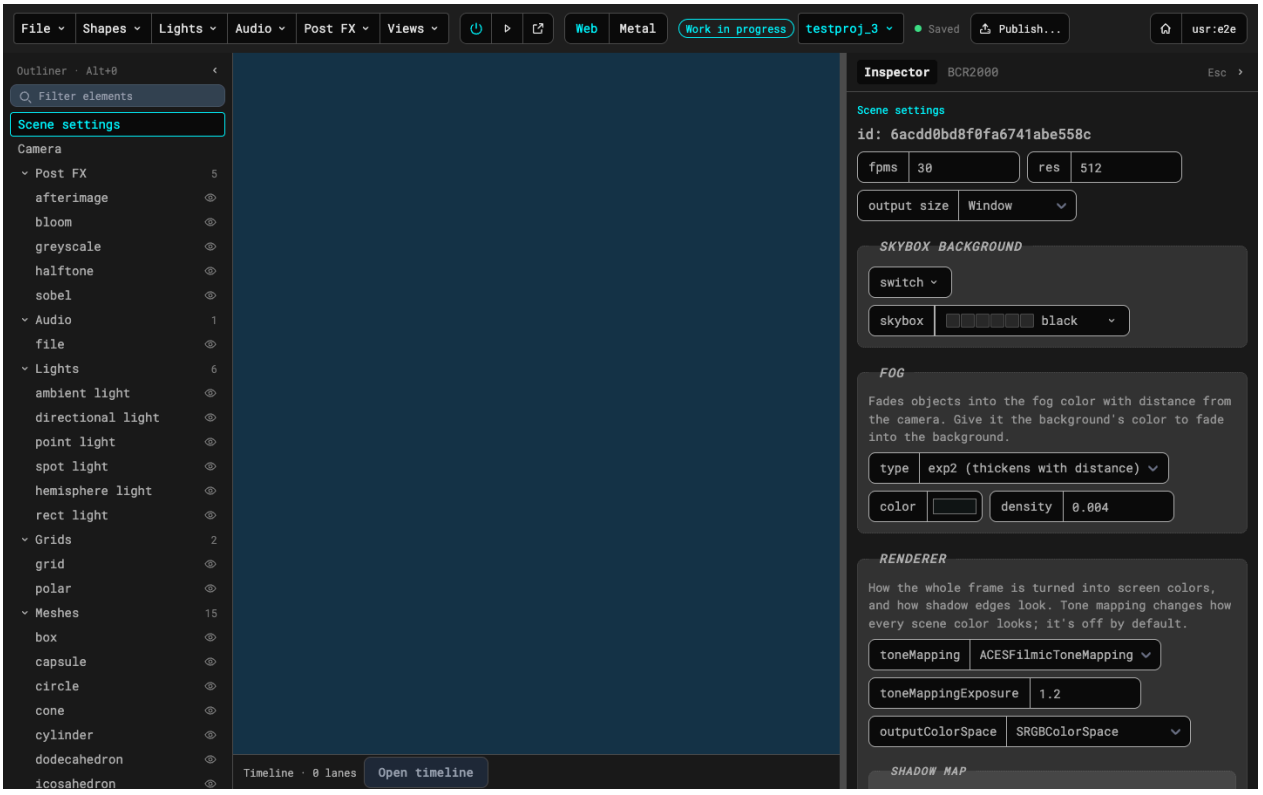
On meshes, OBJs and instance geometries the insert menu adds a motion: rotate, co/sine, orbit or custom function. Meshes and OBJs also offer Positional Audio: file, sine, saw, square, triangle or custom waveform. The audio plays from the element, so it gets louder as the camera comes closer.



Scene settings

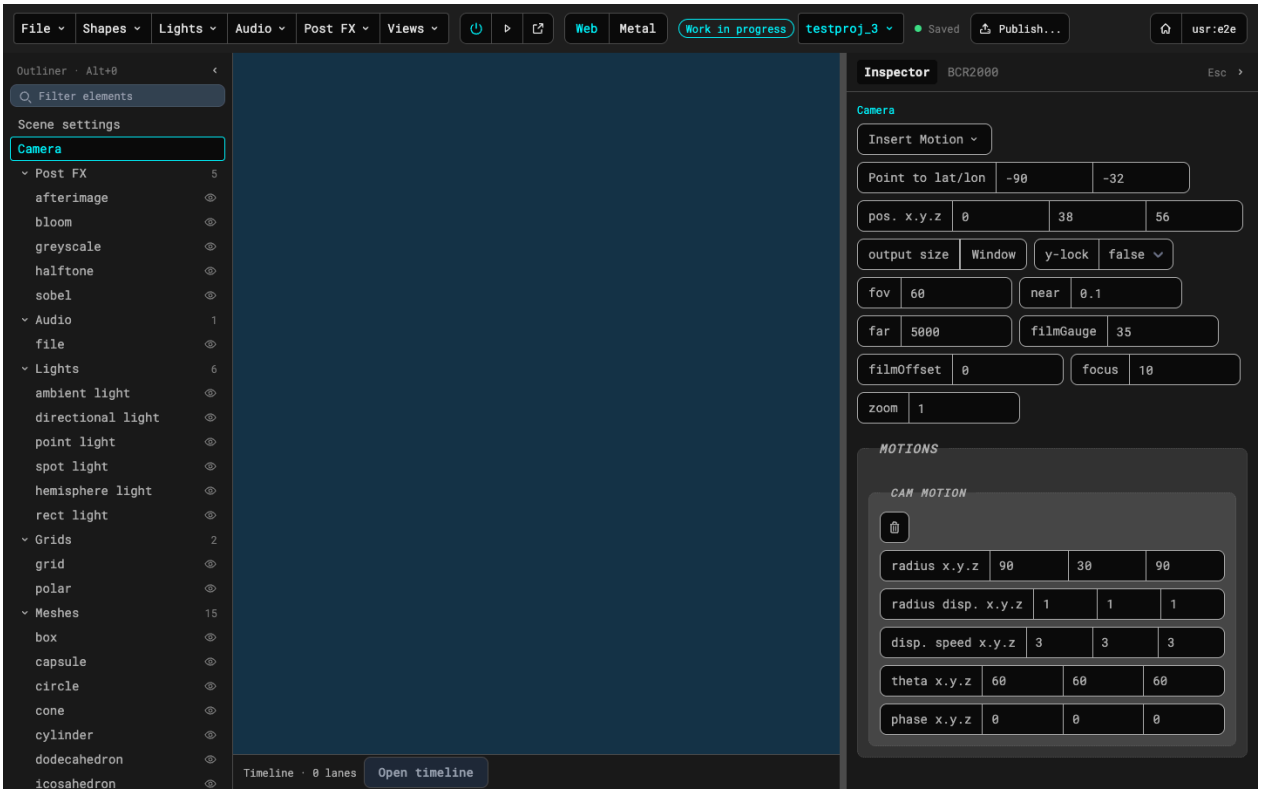
The first row of the outliner. It holds the background (a colour or a skybox; the switch menu changes between them), fog (type, colour and density; give the fog

the colour of the background to fade objects into it), and the renderer (tone mapping, exposure, output colour space and the shadow map type), and the tempo (a BPM from 20 to 300, default 120, that synced video flips follow; a dial can drive it, see Target dials). It also holds the fpms and res settings at the top of the panel, and the output size row below them: Window fills the native runner's window, a ratio takes the largest box of that ratio inside it, and Custom size sets pixels, checked against what the native runner can draw and record (the web preview ignores it).



Camera

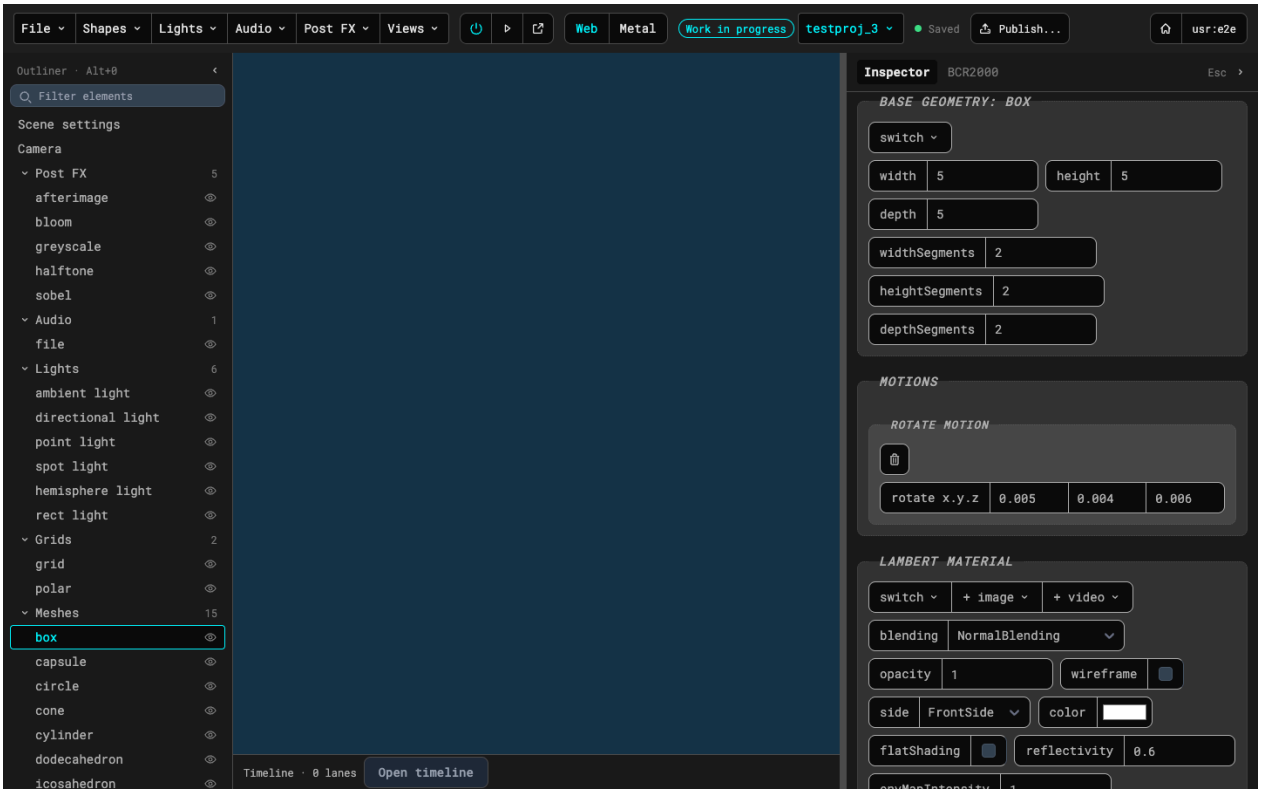
The camera's position and where it points, its lens (fov, near, far, zoom, focus, film gauge and offset), a y-lock, and an output size row that shows the value and opens Scene settings. Insert Motion is a menu that adds a camera motion: Orbit, which moves the camera around the scene, or Focus sweep, which swings the depth of field's focus distance (Swing in scene units either side, Rate in cycles, counted per second or per beat of the scene tempo, and Phase) and never moves the camera. A Focus sweep does nothing in a scene with no Depth of field pass, and the runner's overlay says so.



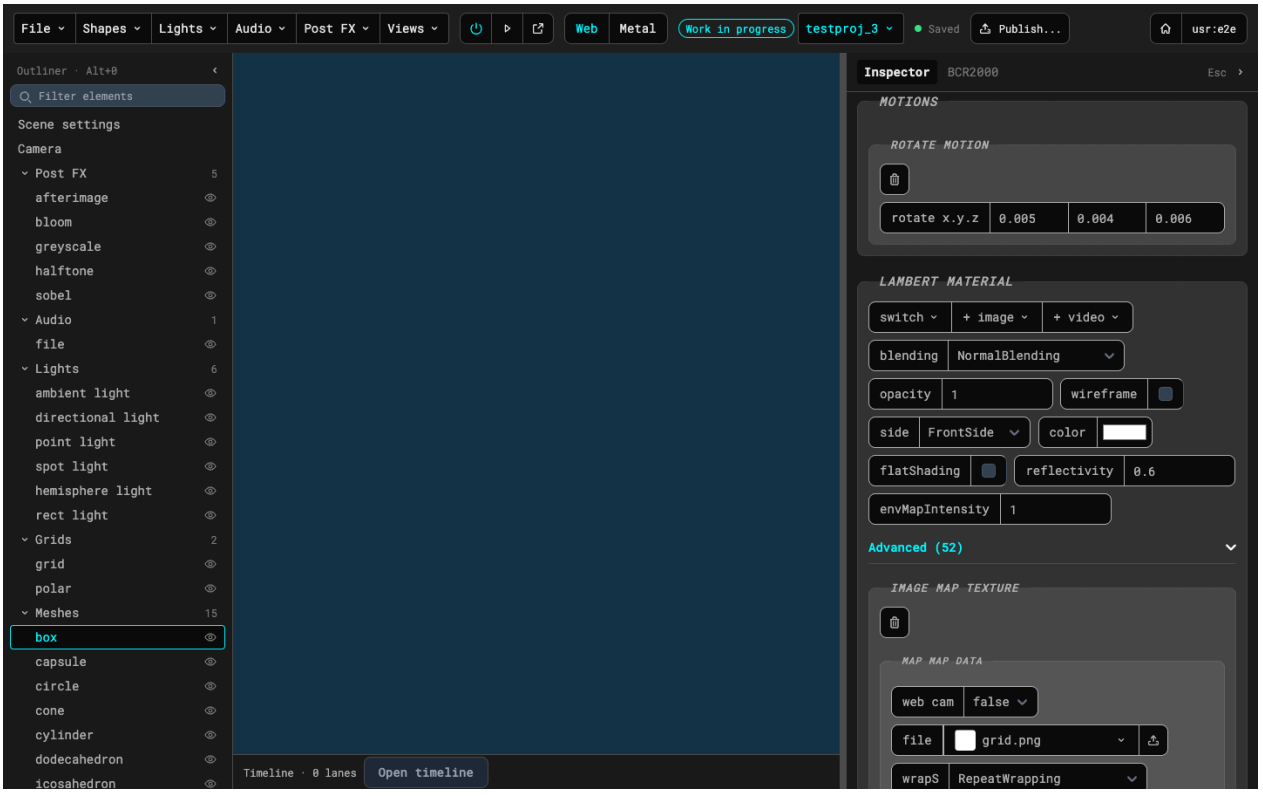
Meshes

A mesh is a geometry with a material, placed in the scene. Its panel has, from the top:

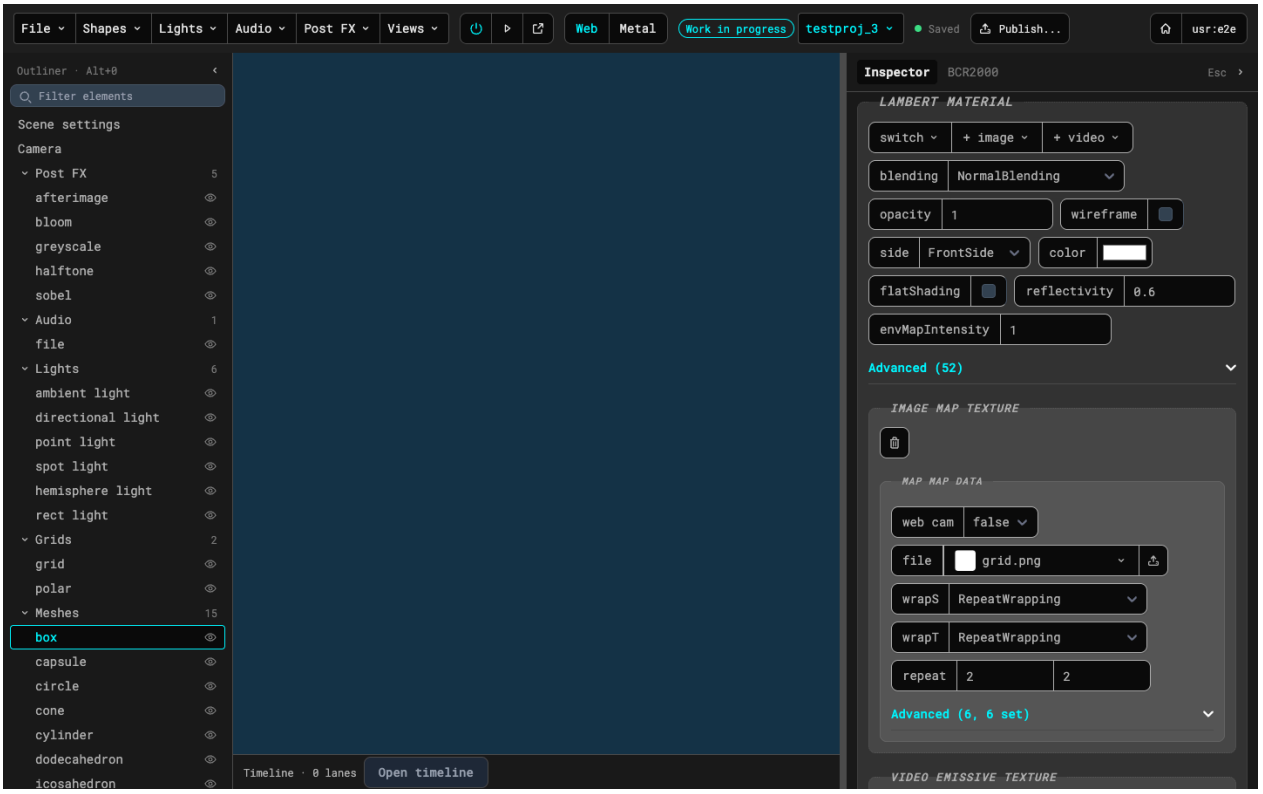
- General settings: scale, position, rotation, and whether it casts and receives shadows.
- Base geometry: the shape's own settings (width, segments and so on). The switch menu changes the shape. A parametric mesh gets its surface from a function, see Functions, Gamepads and BCR2000.



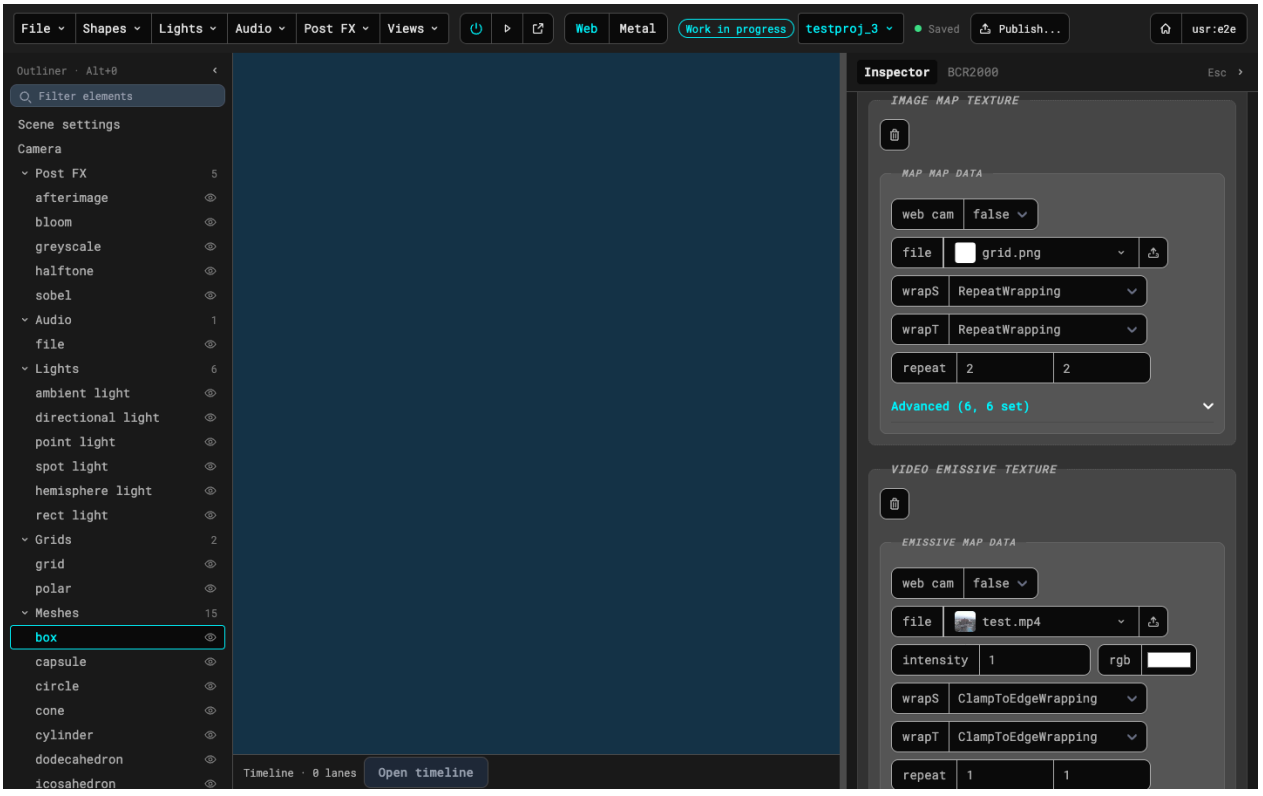
- Motions: each motion is its own group with a remove button. Rotate turns the mesh at a rate per axis, co/sine moves it back and forth, orbit moves it on an ellipse, and custom function calls one of your mesh functions every frame.



- Material: the type (basic, lambert, phong, toon, standard, physical or shader), changed with the switch menu, and the settings that type has. The + image and + video menus add a texture.



- Textures: an image or video on one of the material's map slots. Each has a file field (see Media), how it wraps and repeats, and an Advanced section. A video texture also has volume, speed, a start (s) and loop. The start is how many seconds into the clip the video begins (a start past the clip's length wraps around it); it then loops from the beginning, so two copies of one clip stay the same distance apart. A webcam has no start.



Video flip. A material takes a second video on its colour map: + video › map twice (a third is refused, and a second image on the colour map is still refused). With two video maps the material shows a Video flip section. Switch flip on and the picture flips by itself between the two videos, A (the first) and B (the second), and crossfades. Put the same clip on both maps and give B a start of a few seconds, and the figure is in a different place after each flip. The settings:

- **sync** is free or scene tempo. Free flips at its own rate in milliseconds (30 to 10000). Scene tempo flips every so many beats (1/8, 1/6, 1/4, 1/3, 1/2, 2/3, 1, 4/3, 2 or 4) of the BPM in Scene settings; the 1/6, 1/3, 2/3 and 4/3 are triplets, and a triplet flip still lands on the beat every 1, 2 or 4 beats; all the elements that are synced flip together, and changing the BPM speeds them up without a jump.
- **crossfade %** (0 to 100) is the share of each flip that is spent fading. 0 is a hard cut back and forth, 100 fades all the time, 50 fades for the first half of each flip and then holds the picture.

The flip works on meshes, on OBJs and on instance geometries. The sound follows the picture: each video's volume is its own volume times its share of the mix, so a cut switches the sound and a crossfade fades it. While a picture is fully showing, the other video is not drawn at all. A flip with fewer than two video maps says so in the runner's overlay and the element draws as usual. The native runner does not do the flip yet: it plays these scenes without the flip.

On a shader material the first video's `tex<N>` holds the mix and the second video's `tex<N>` stays the plain video, so the shader needn't do anything. The runner also sets a `flipMix` uniform, a float from 0 (A) to 1 (B) that is the live mix, which the shader can use for its own transition between two textures. A shader that doesn't declare it ignores it, and the editor doesn't list `flipMix` as a uniform of yours:

```glsl
uniform sampler2D tex0; // video A
uniform sampler2D tex1; // video B, plain
uniform float flipMix;
// a wipe from A to B instead of a fade
gl_FragColor = vUv.x < flipMix ? texture2D(tex1, vUv) : texture2D(tex0,
```

The two videos are separate files in memory even when they are the same clip, so a flipping element costs two video decodes. To flip a clip against itself backwards,

make a reversed copy beforehand and use it as the second map: `ffmpeg -i clip.mp4 -vf reverse -af areverse clip-rev.mp4`. Browsers cannot play a video backwards.

Key out black. A video texture (a file or a webcam) can drop its black out of the picture, so what is behind it shows through. Set **key out black** to true on the texture and two rows appear:

- **threshold** (0 to 1, default 0.1) is the brightness at and below which a pixel is cut. It is measured as luma, the picture's own brightness (0 is black, 1 is white), so it is the same whatever the codec: camera footage and compressed clips are rarely pure black, more like 0.06 to 0.08, which a threshold of 0.1 catches. Raise it until the black is gone, and lower it if dark parts of the picture start to vanish.
- **softness** (0 to 0.5, default 0.05) is how far above the threshold the edge fades in instead of cutting. 0 is a hard edge; a few hundredths hides the noise in dark footage.

The key reads the video before anything tints it, so a red material colour or a collider colour way with a zero channel colours the picture and doesn't cut it out, and saturated reds, blues and yellows in the footage stay. A keyed pixel is cut out completely: it hides nothing behind it, so instances behind a keyed area of a collider grid show through from every angle. The soft edge is a smooth fade where the material is drawn blended (an opacity below 1 in a scene without post-processing passes) and a fine grain everywhere else, including every collider and any scene with a post pass, which looks like a fade from a normal distance. A material whose opacity dial first drops below 1 changes from grain to fade as it does. With a video flip, the first video's key applies to the mix. On a shader material every keyed video is keyed automatically wherever the shader reads it as `texture2D(tex0, ...)`, with no change to the saved shader (a shader that hands the sampler to a function of its own isn't keyed), and the fragment is cut or faded by the darkest keyed sample it took. The key applies to the colour map of basic, lambert, phong and standard materials; toon and physical materials have no map here. A keyed-out area of a built-in material or a collider casts no shadow; a shader material keeps its full shadow. The Mac runner keys the same way, shadows and shader materials included (a package published before the Mac runner could key needs publishing again).

A shader material shows its vertex and fragment shader as code to edit.

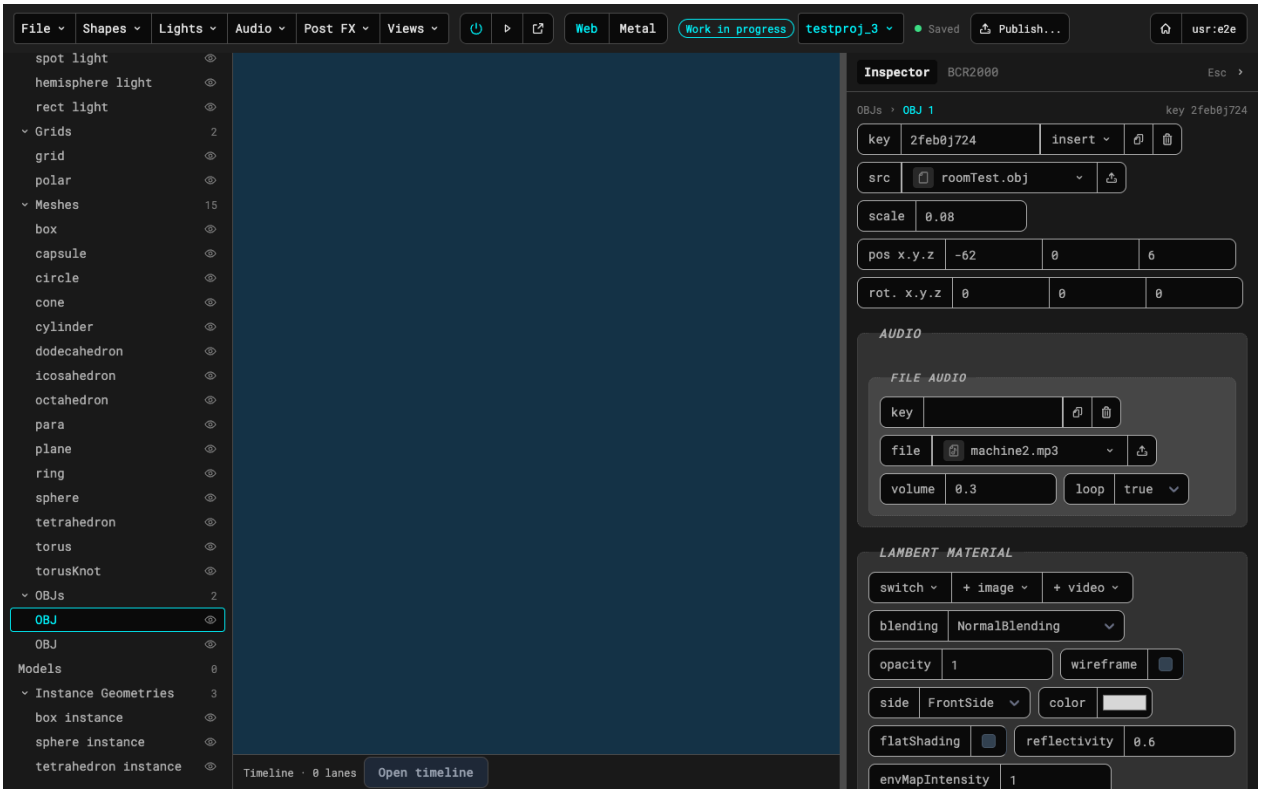
Shader helper. The Shader helper button above a shader material's code opens an assistant. Type a question and press Send; the reply streams in, with a Stop button. The current vertex and fragment code, the uniforms and the textures always go with the question, so you can ask for a new shader ("a slow scrolling stripe pattern using tex0 with a speed uniform"), or work on the one that is there: what does this do, make it twice as slow, why does it look wrong, make this old scene shader work in this runner, or add range comments to its uniforms. An answer that only explains has no code. When it changes code, each vertex or fragment block is shown in a side-by-side diff against the code you sent, so you can read the change first. Apply to vertex or Apply to fragment (whichever the block is) replaces that field with the block, Apply both does both at once, and Save as new library shader asks for a title and adds the block to your shader library. Nothing changes in the scene until you press an Apply button. The helper knows the runner (GLSL ES 1.0, `texture2D`, the `time`, `resolution` and `tex0` uniforms, the range comment format) and your BCR2000 setup: it writes a few scalar uniforms with ranges worth performing, lists them as dial-ready and names free encoders of your controllers, but it never maps a dial for you. It needs your key on the Profile page; the modal says so with a link when there is none. If the runner cannot compile the shader of a mesh, an OBJ or an instance geometry, a Fix this error button appears next to Shader helper; it opens the helper with the compile log in the question. The server settings are `DST_SECRETS_KEY`, `DST_AI_MODEL` and `DST_AI_MODE` (see the README).

Suggested ranges and descriptions. Older shaders (Times Squares v26, the amorphics scenes) have uniforms with only a starting value, so nothing says how far a slider or a dial should go. When some uniform of a shader material has no `min`, no `max` or no `desc` in its Uniforms JSON, the Uniforms section shows Suggest ranges and descriptions (with Shader analysis on Manual; on Auto the same request goes out by itself). The helper reads both shaders, works out from how the code uses each of those uniforms a sensible min and max (for numbers, ints and vectors) and a one-line description of what changing it does to the picture, and lists them under the rows as "Suggested by the helper". Nothing is written until you choose: Accept writes one uniform's suggestion, Accept all writes the whole list, and Dismiss throws it away. An Accept only fills what is still missing at that moment, so a value, a range or a description you set yourself is never overwritten.

The result is kept in the Uniforms JSON as `{ "value": 1.5, "min": 0, "max": 5, "desc": "How fast the stripes scroll" }`; the `desc` is shown as a small line under the uniform's row and as a tooltip on its name, the dials read only `min` and `max`, and the runner ignores `desc`. Colours, bools and matrices get a description only. To change a `desc` afterwards, edit the Uniforms JSON. Without a saved key the button is disabled and links to the Profile page, and Auto never asks.

OBJs

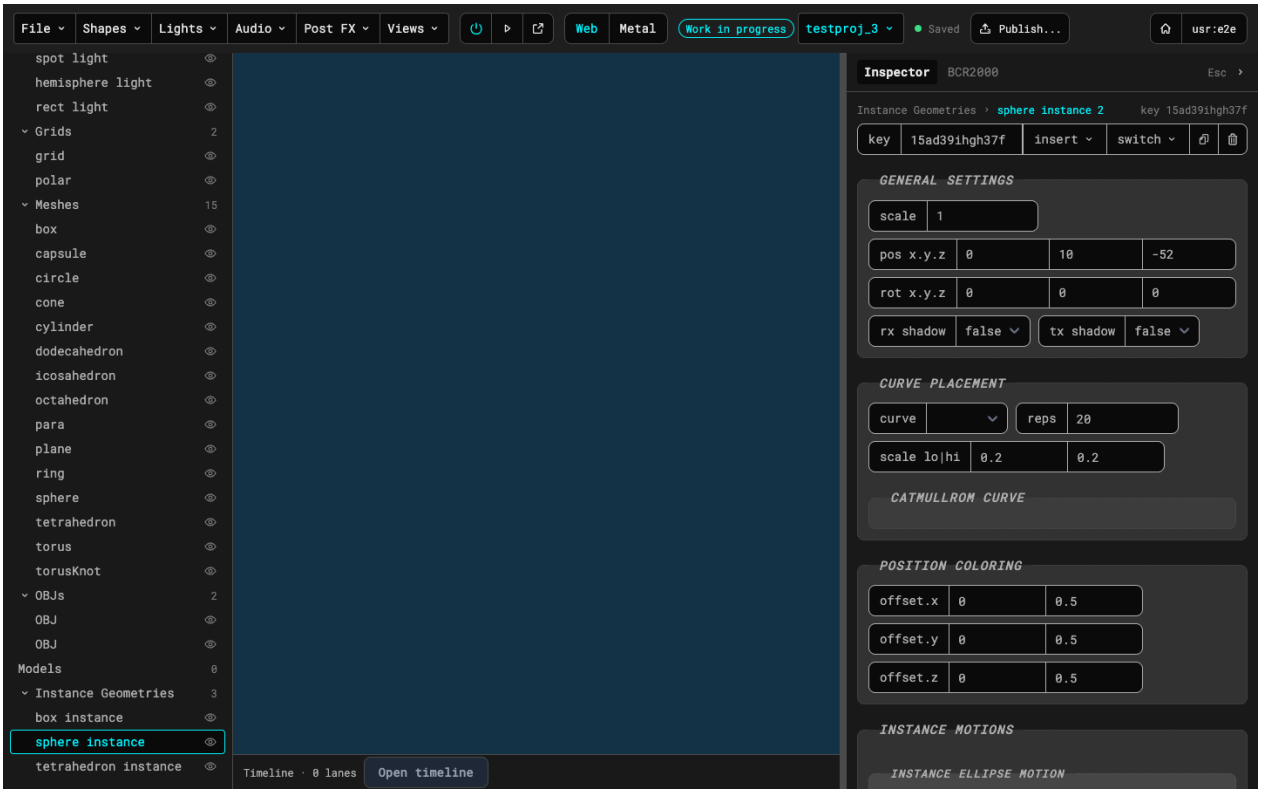
An OBJ is a 3D model loaded from a file. Pick the file in the src field, then set scale, position and rotation. One material is used for every part of the model. The insert menu adds a motion or positional audio.



Instance geometries

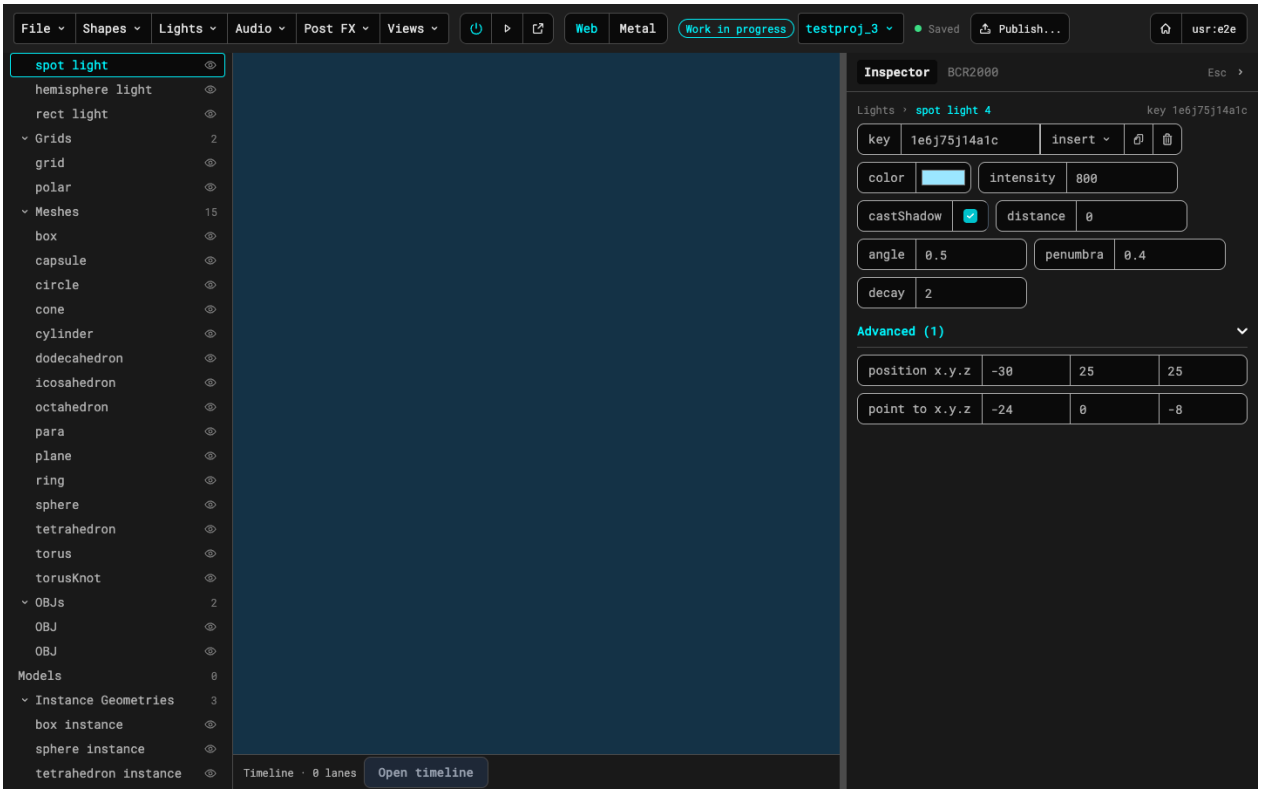
An instance geometry (also called a collider) draws one shape many times. Its panel has the general settings, a placement (how the copies are laid out: in an

array, along a curve, or geometrically), colouring for the copies, the shape's geometry and material, and instance motions that move every copy. When the material is a shader, the panel also has a Shader uv setting: "across grid" (the default) spreads the shader over the whole grid, each copy showing its slice, the way a texture or video spreads over a collider, and the slices stay put while the copies move; "per instance" gives every copy the shader's own uv, so every copy shows the whole shader. A dial on a shader uniform changes all the copies in both.



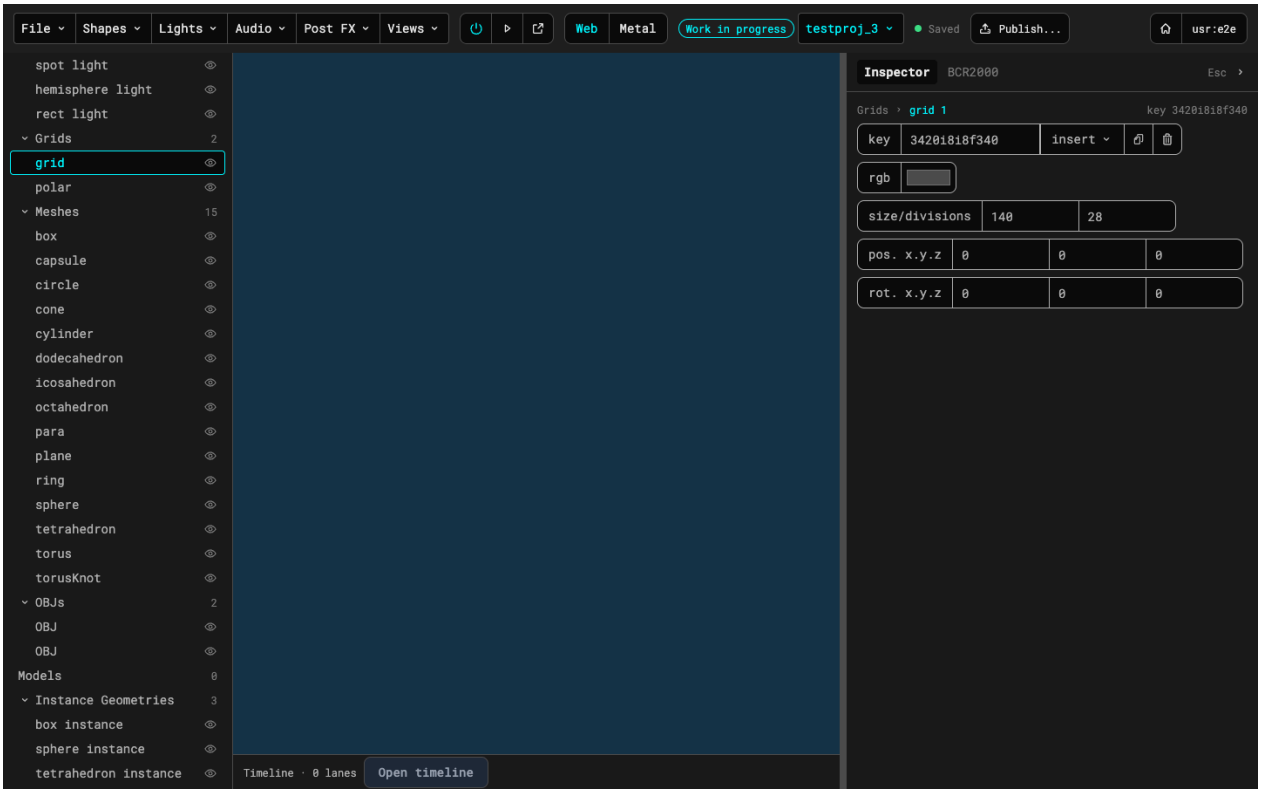
Lights

Each light has a colour, an intensity and the settings its kind needs: a spot light has distance, angle, penumbra and decay, and Advanced holds its position and the point it aims at.



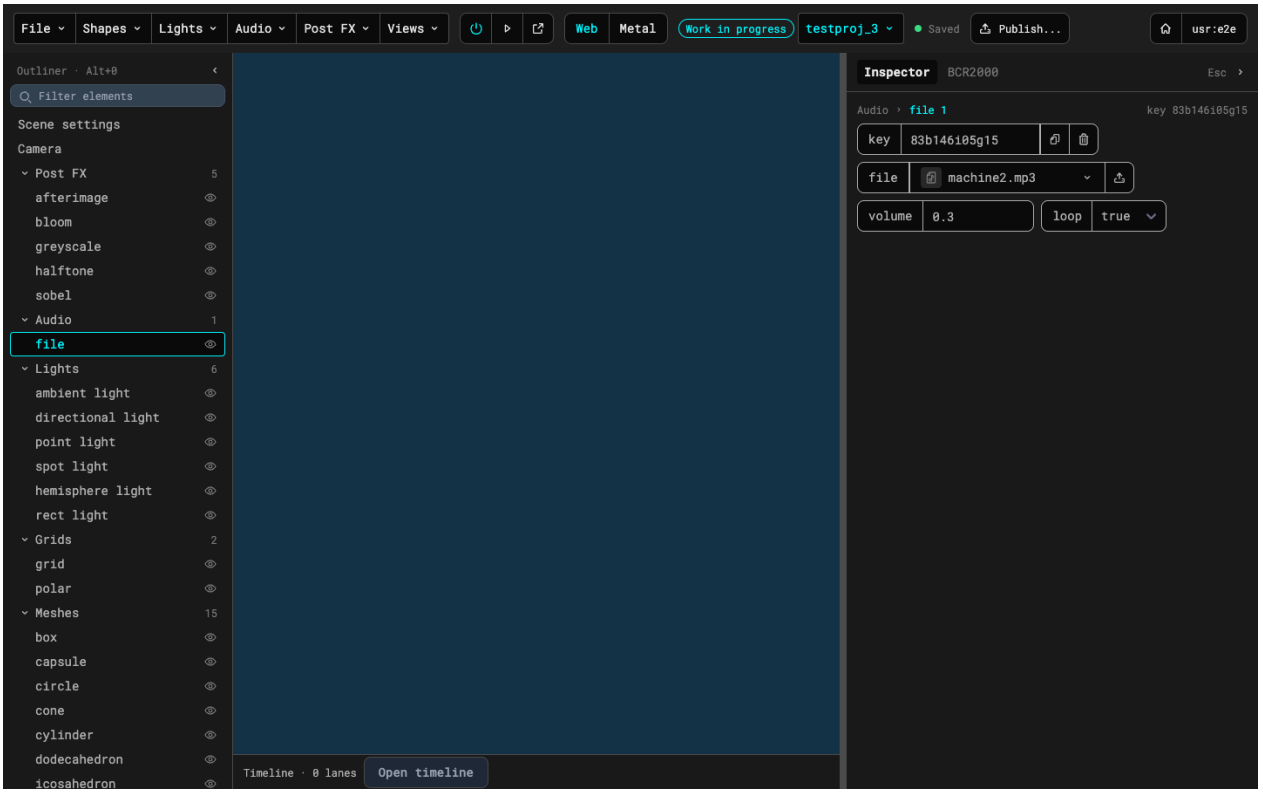
Grids

A grid is a flat reference grid, or a polar grid for the circular one. Set its colour, its size and number of divisions, and its position and rotation.



Audio

A scene-level sound. A file entry plays a sound file with a volume and a loop setting. Oscillators are positional audio: add them from a mesh's or OBJ's insert menu.

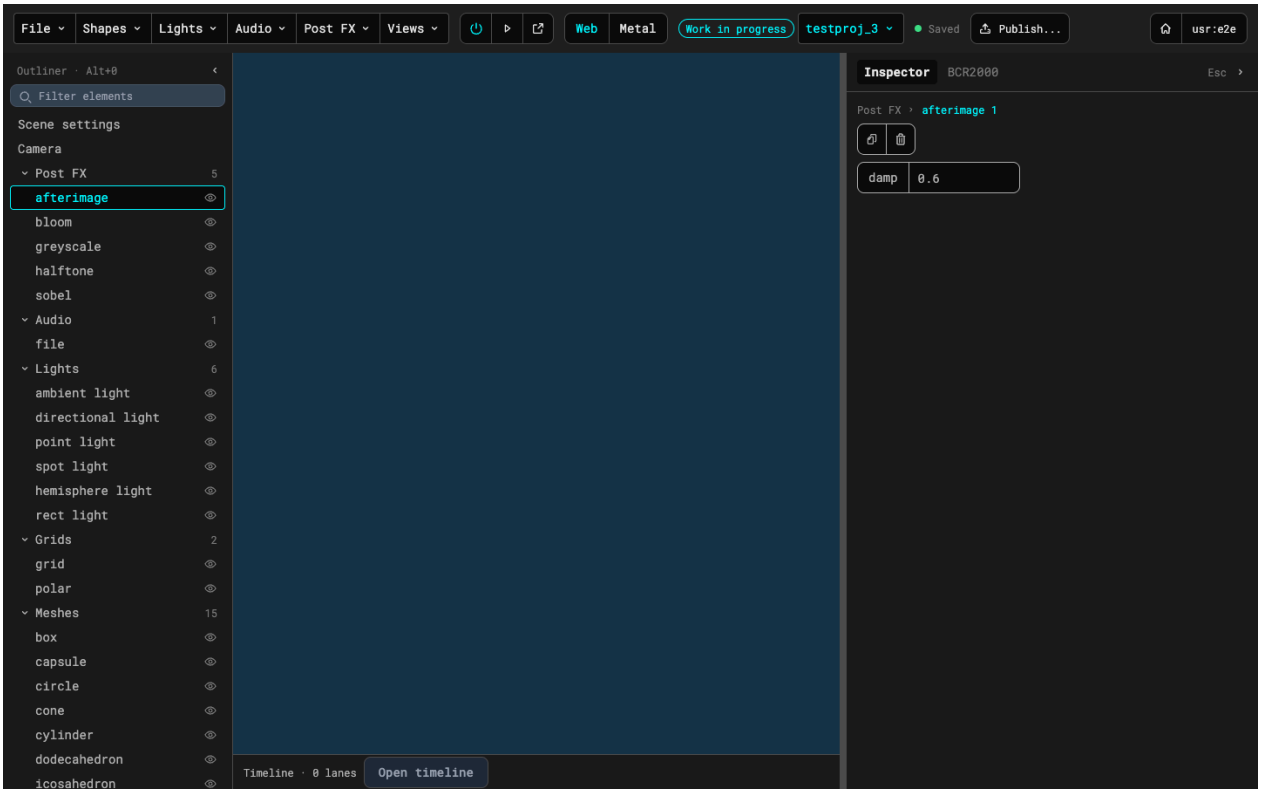


Post effects

Each effect has its own few settings, for example damp on afterimage. Duplicate an effect with the copy button at the top of its panel (the copy goes to the end of the list and is selected), and delete it with the trash button next to it.

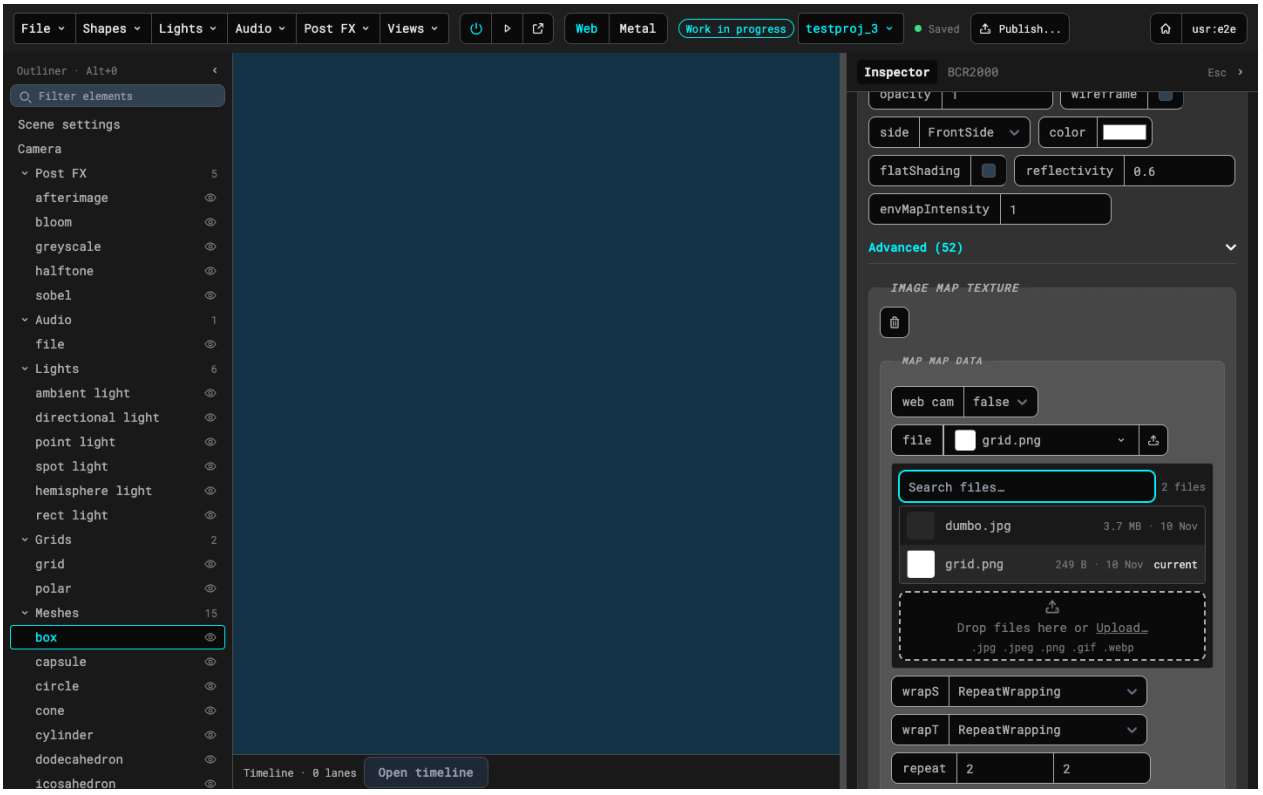
The effects run from top to bottom in the Post FX list, so the order changes the look. To reorder them, drag an effect's row in the outliner onto the row where it should go (the dragged row dims and a line shows where it lands), or select it and press Alt+Up or Alt+Down. The move is an ordinary edit: the preview redraws in the new order and it is saved with the next save.

A target dial on an effect type, such as scene.post.bloom, drives the first effect of that type. If a scene has two effects of one type, reordering them changes which one the dial drives, and the effect's panel says which one it is.



Media

Fields that use a file (a texture's image or video, an OBJ's src, a sound, a skybox) are media pickers. The field shows the current file. Click it to open the list of that kind of file just below, with a search box, the size and date of each file, and a drop zone.



- **Pick** a file by clicking it. The list closes and the field shows the name.
- **Search** by typing in the box at the top of the list.
- **Upload** with the button at the end of the field, or drop files on the drop zone. Uploads show their progress and can be cancelled. A file can't be uploaded if its type doesn't fit the field (images take jpg, jpeg, png, gif and webp; videos take mp4, mov, webm and m4v; sounds take mp3, wav, ogg, m4a and aac; OBJs take obj and mtl) or if a file with that name is already there.
- **Skyboxes** are six images that make a cube around the scene. The skybox field lists the skyboxes already in the media folder, and Add skybox... makes a new one from a name and its six faces.
- **Missing files:** when a scene names a file the media folder doesn't have, the field shows a red notice with Upload it... and Pick another. Nothing changes until you choose one.

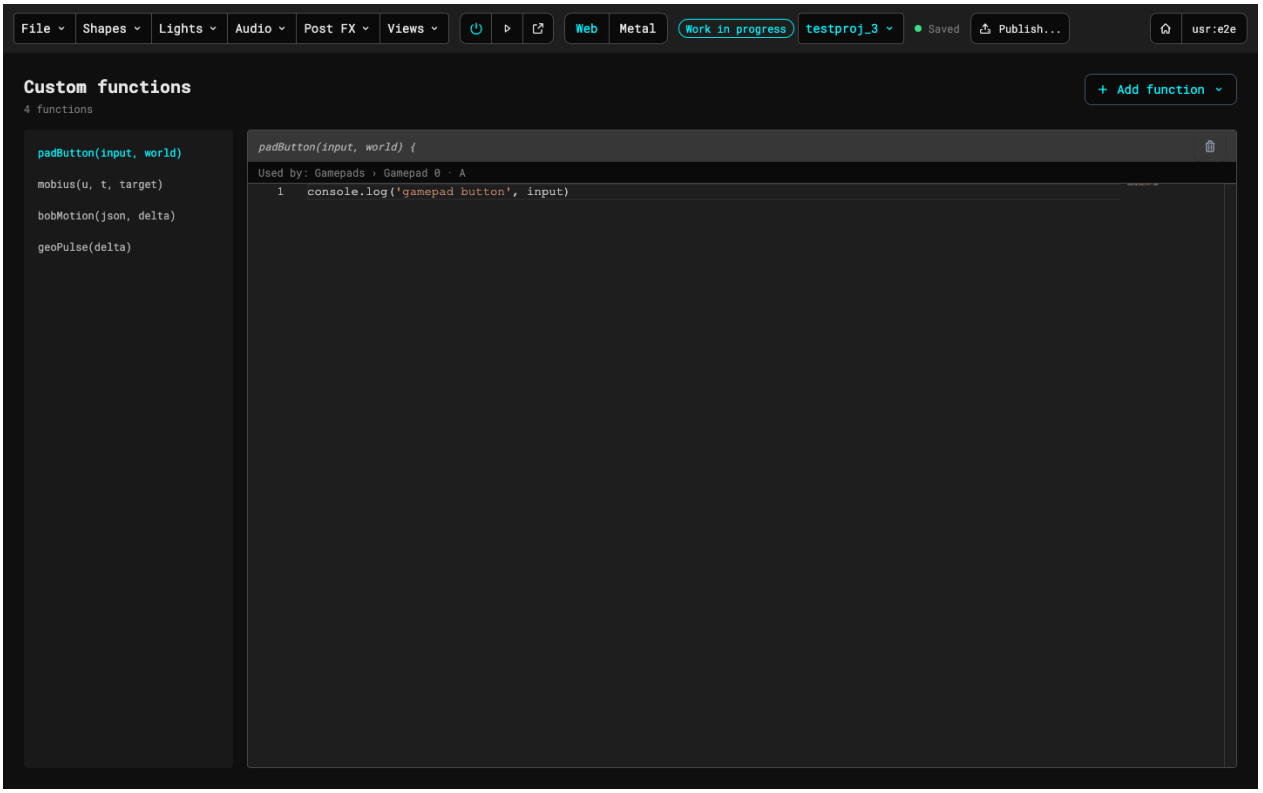
Functions, Gamepads and BCR2000

Open these from the Views menu, or with Alt+2 to Alt+4 (Option on a Mac). Alt+1 goes back to the scene editor.

Functions

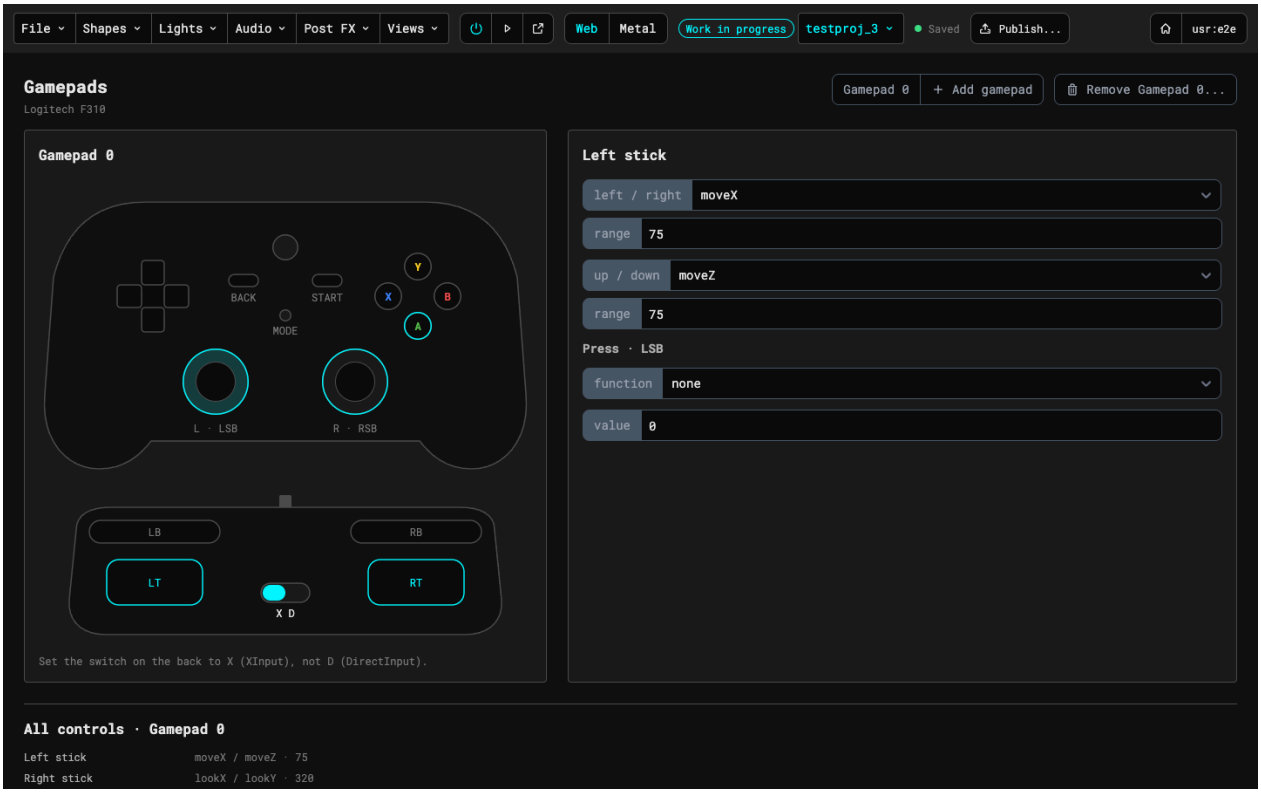
Custom code for the open version. The list on the left shows each function's name and arguments. Pick one to edit its code in the editor on the right; a line under its title says where it's used (for example, a gamepad button). The trash button removes it. Add function creates a new one: pick Create..., then give it a name and a type, or start from the klein, mobius or mobius3D presets.

The types are ctrl (a control function, called by a gamepad button or a BCR2000 dial), para (a surface for a Parametric mesh), mesh (a function that moves a mesh) and geo (a function that changes a geometry).



Gamepads

Sets up Logitech F310 gamepads. Set the switch on the back of the pad to X, not D. Click a stick, trigger or button on the picture of the pad, then set what it does in the panel on the right. A stick takes an action and a range for each direction; pressing a stick or a button can call a function with a value. The buttons at the top right switch between pads (Gamepad 0, 1 and so on), add another pad and remove the open one. The list at the bottom shows every control of the pad at a glance.



BCR2000

Maps the controls of one or more Behringer BCR2000s (or BCR32s) for the open version. The heading counts the dials and controllers. Next to it, Save knob positions as start values (⌥⇧S on a Mac, Alt+Shift+S on Windows) takes the knobs' positions as the version's start values, as an unsaved edit until you save the version, and Recall start values (⌥R, Alt+R) sends them back to the knobs. A dot beside them says which runner is listening: Preview open, Runner window open, or no runner open, in which case Recall is off until you start the preview or open it in a

new window. Sync from BCR, a row of the gear menu, asks the BCR for its own preset (it needs the runner open and a BCR connected). The gear button (the cog) opens a menu in the style of GitHub's View menu, with a row for MIDI ports, Controllers and Remove BCR2000 mapping, each with its current value and a chevron; a row opens its own dialog. Remove BCR2000 mapping asks first and then removes everything.

Controllers. The Controllers row of the gear menu (its value is the number of controllers) opens a dialog with a row for each controller: its name, a MIDI channel and a model (BCR2000 or BCR32). Set the same channel in the BCR's own preset. A change applies as you make it, as an unsaved edit of the version; a channel another controller uses is refused under its row. Add controller adds another on the lowest free channel, and Remove asks first and says how many dials and buttons move to the first controller and how many buttons are dropped. Done closes the dialog. Every controller has its own grid below, side by side; with three or more the row scrolls sideways.

Other controllers, MIDI learn and profiles. A controller reads either the BCR's 14-bit pairs (input "BCR (14-bit pairs)", the default) or plain 7-bit messages ("Plain 7-bit CC"): every CC 0 to 127 is one complete message, a button is any note or any CC you gave it, and a dial on it can use CC 0 to 127 (a BCR's stay 0 to 31). The Controllers dialog sets the input and, for a 7-bit controller, the port: part of the name of the MIDI output (any case) that its start values are sent to as one CC per dial; leave it empty and nothing is sent. Each controller needs its own MIDI channel.

The gear menu's **MIDI learn** row (it needs the preview or a runner window open, and Build mode) opens a dialog. Move a control on your controller and the dialog shows what it heard ("CC 77 · channel 9 · Test Controller Port 1"); move another and it replaces it. Click a parameter in the list and the control is linked: a channel no controller has gets a new 7-bit controller named from the port, and the control becomes a target dial with that parameter's range. A note, or a CC with "This is a button" ticked, shows the button actions instead (Next scene, Previous scene, Recall start values, Record performance) and takes the lowest free key, control 1 to 64. The dialog says what a link replaces, and stays open for the next one. If a CC arrives on a BCR channel and no knob turn follows within 150 ms, the dialog offers

to switch that controller to plain 7-bit CC. A 7-bit controller's grid shows only its own dials and buttons, in rows of 8, with each dial's CC and channel.

The **Controller profiles** row opens a dialog with the profiles the editor ships (the BCR2000) and the ones saved for everyone in your group. A profile describes a device: its name, maker, port and units (name, channel, model, input, and its labelled dials and buttons), never what a control is linked to. Apply copies the profile's units into the scene as controllers and is refused, changing nothing, if a channel is already used; a scene that is still the untouched default BCR is replaced instead. Export downloads the profile as JSON, Import reads a file (refused with the reason when it is too large, not JSON or not a profile) under a name you can change, and Save as profile makes one from a controller of the scene. Deleting a profile never changes a scene.

MIDI ports. Normally the editor finds a BCR by the word BCR in the name of its MIDI port. When a BCR is connected through a MIDI interface, its ports have the interface's names (for example "MIDIFlex4 Port 3"), so nothing is found. The MIDI ports row of the gear menu (its value reads "auto", or how many inputs and outputs you chose, like "2 in, 2 out") opens a dialog that lists every MIDI input and output the runner sees; the row is off until a runner (the preview or its own window) is open. Tick the outputs the BCR is connected to (start values, the LED rings and Sync from BCR's preset read are sent there) and the inputs it comes in on (the preset read listens there, and the status counts them), then press Done. Knobs and buttons are still read from every input. With nothing ticked, ports named BCR are used as before. The choice is kept in this browser. If the dialog says the runner hasn't reported its ports, reload the runner window. Two units on one setup still need different MIDI channels in their presets, because start values go to every ticked output.

Dials. A controller has 32 dials (MIDI controls 0 to 31). The grid is the same one the manual show uses. Each dial shows the knob's live value large in a ring, its label under it, and a small encoder number (enc 1 to 56) under that. The encoder number follows the usual BCR layout, so it matches the unit only when the preset does. The dial doesn't show its CC or NRPN: hover it to read where it is and what it sends. Click a dial to open its dialog, the same one the sequence page uses. The line at the top of the dialog says the same, for example "Top row, group 4, knob 1 ·

encoder 25 · NRPN 25 · channel 1". A dial shows an arc for its live value as soon as its knob moves, in grey until the dial is mapped, and an unmapped dial has no start tick. The dialog sets what the dial controls: the controller it is on (with two or more controllers, saving moves the dial with its function and start value), a target (an element chosen by its key, see Target dials below), a label, an input range and an output range, the NRPN number for groups 2 to 4, and a function that runs with the dial's value. Reset to default puts the defaults into the form, and nothing changes until you press Save; Cancel or Esc closes the dialog without changing anything, and Remove mapping clears the dial. On this page Save is an unsaved edit of the open version, like the other changes here. A dial's start value isn't in the dialog: use Save knob positions as start values for all of them at once. A dial with settings is lit. With a runner open, a mapped dial shows the knob's live value as an arc and its start value as a white tick, and turns amber while the knob is off its start value. The input range is what the dial sends: 0 to 1000 for the usual 14-bit knob, so a value of 500 is halfway.

Target dials. A dial whose type is target drives numbers in the scene without a function. Choose the Target key (an element, or Camera, Fog, Renderer, Tempo, or a post-processing pass of the scene) and the Settings select under it lists what that target offers: for an element its position, rotation and scale, its material's or light's own settings (opacity, roughness, intensity and so on) and the numbers of its animations (a rotate's speeds, an ellipse's radii and speed; a collider also offers the base numbers of its per-instance animations, not their random ranges); for the camera its lens (field of view, zoom, near, far, focus) and the numbers of its camera motions (an orbit's radii and speeds, a focus sweep's swing, rate and phase); for the fog, the renderer and a post pass their own numbers; an element whose video flip is on offers Flip rate (ms, 30 to 10000) and Crossfade % (0 to 100); an element with a keyed video texture offers Key threshold (0 to 1) and Key softness (0 to 0.5), which apply to every keyed texture of its material; and the Tempo target offers BPM (20 to 300). A rate dial on an element that is synced to the tempo changes the saved rate with no visible effect until the element is set to free. You can choose several settings, and a turn of the knob sets all of them to the same value. One dial has one output range, so the Settings select only lets you pick settings with the same range: choose Scale x and Scale y and z stay available, but a setting that runs from 0 to 1 is greyed out with its range beside it, and clearing your choices frees

them all again. The first setting you choose fills the output range from that setting's limits (a light's intensity, which has no top, runs from 0 to twice its current value); you can edit the range afterwards. A target dial needs an element and at least one setting before it can be saved. The runner sets the settings as the knob turns and, like a function dial, applies the dial's start value when a scene loads and on Recall. If the element isn't in the scene, or the setting isn't there (for example fog in a scene without fog), the dial does nothing and the runner's overlay says so once. The animations change while they run, not in the saved scene: when the scene reloads they start from their saved numbers again. The camera spin's turn speed is in degrees.

Shader uniforms and time on a target dial. An element whose material is a shader (a mesh, an OBJ or an instance geometry) also offers the shader's uniforms in the Settings select: one "Uniform <name>" for each float, int or bool the vertex and fragment shaders declare, and one per part of a vector ("Uniform color x", y, z, w). Matrices, arrays, textures and the runner's own `time`, `resolution` and `tex0`, `tex1` ... are not listed. A uniform's range is the one the Uniforms tab of the shader shows: its min and max, else the numbers in the shader's comment (`uniform float xB; // 0 5`), else 0 to twice its current value (0 to 1 when that is 0); the parts of a colour and a bool run 0 to 1, and an int moves in whole steps. Because the range is the same, the red, green and blue of a colour (or three separate `red`, `green` and `blue` floats with the same range) can be chosen together and one dial sets all of them. A uniform that the Uniforms JSON lacks is listed too, and starts at 0. An int is rounded, and a bool is on when the dial's value is above the middle of its output range. "Time rate" and "Time offset" drive the shader's clock: the rate scales how fast time runs (0 to twice the saved rate, or its min and max), changing it while the scene runs doesn't make the animation jump, and the offset (0 to 10, or the min and max of the Time offset row) is added to time and replaces the saved offset while the scene runs. Neither is saved in the scene; when it reloads the saved rate and offset apply again.

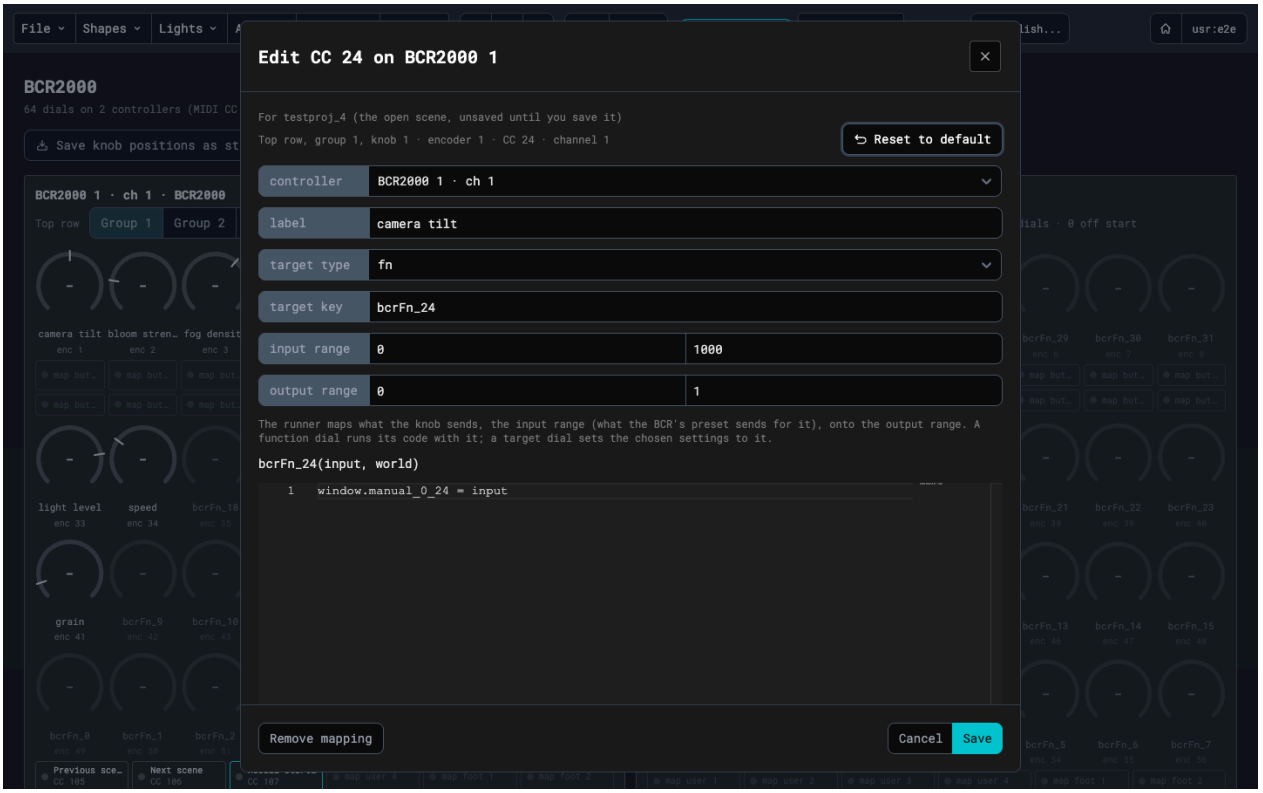
OBJs and instance geometries also offer their material's own numbers (opacity and the rest for the material type) like a mesh does, so a dial can fade them.

Sync from BCR. The gear menu's Sync from BCR row opens a dialog that reads the BCR's preset as soon as it opens (a real unit takes 10 to 15 seconds, so it says

Reading...) and then shows one review: the presets found, the input ranges that differ from the dials' (for example "CC 7: preset sends 0–16383, scene says 0–1000"), the NRPN numbers of groups 2 to 4 that differ ("group 2 knob 1 (CC 24): NRPN 9 to 100"), the encoders in groups 2 to 4 that don't send NRPN, and what the editor can't use (7-bit and relative encoders, a controller the preset has nothing for). Apply sets the ranges and the numbers together, as an unsaved edit of the version here and as a saved change to the playing scene on the sequence page; Cancel changes nothing. If the preset agrees with the dials, the review says so and Apply is off. If the BCR doesn't answer, SysEx is blocked or no runner is open, the dialog says why, with Retry.

Groups. The push-encoder row (CC 24 to 31) has four groups, the BCR's four banks. Group 1 sends the normal control change; groups 2 to 4 send NRPN, so each of those dials has an NRPN number that has to match the number in the BCR's preset. A new dial starts with its encoder number (encoder 25 sends NRPN 25), so a preset that follows that convention needs no sync; dials already saved keep their numbers. The Group 1 to 4 tabs in each controller's grid switch which group its top row shows, and a group with dials in it is marked. The tabs follow the knobs too, since the BCR's group keys send no MIDI: turn a top-row knob of another group and the tab and the top row move to that group (a click on a tab holds until the next such turn). The numbers are an unsaved edit until you save the version.

Buttons. Click a button to open its dialog. Buttons 1 to 16, the user keys and the two footswitches map the same way as on the sequence page, with Learn to fill in the number from a press of the button: a CC (64 to 127) or a note, momentary or toggle, and an action (run a function, previous or next scene, recall the start values). CC 98 and 99 can't be used for a button, since the runner reads those as the start of an NRPN message.

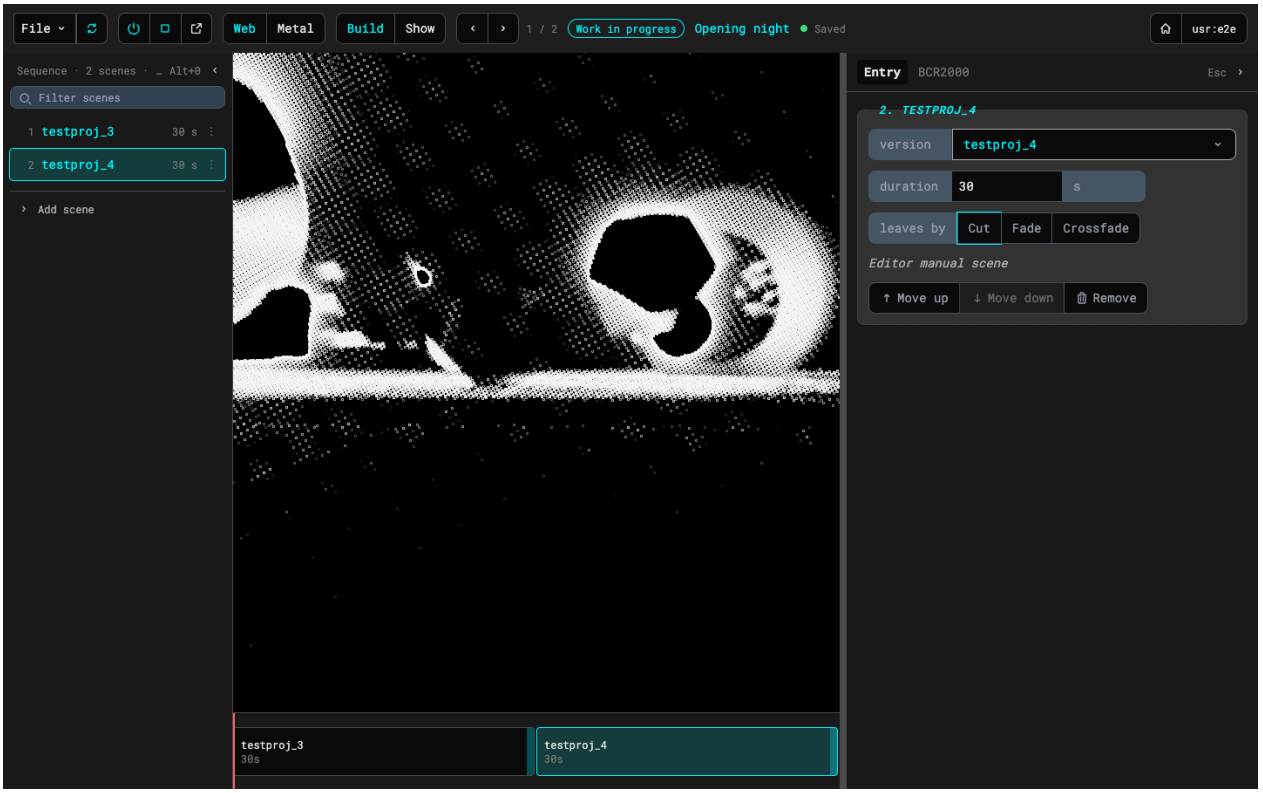

File
Shapes
Lights
usr:e2e
BCR2000
64 dials on 2 controllers (MIDI CC
Save knob positions as st
BCR2000 1 · ch 1 · BCR2000
Top row
Group 1
Group 2
camera tilt
bloom stren…
fog densit
light level
speed
grain
Previous sce…
CC 105
Next scene
CC 106
Edit CC 24 on BCR2000 1
For testproj_4 (the open scene, unsaved until you save it)
Top row, group 1, knob 1 · encoder 1 · CC 24 · channel 1
Reset to default
controller
BCR2000 1 · ch 1
label
camera tilt
target type
fn
target key
bcrFn_24
input range
0
1000
output range
0
1
The runner maps what the knob sends, the input range (what the BCR's preset sends for it), onto the output range. A function dial runs its code with it; a target dial sets the chosen settings to it.
bcrFn_24(input, world)
1 window.manual_0_24 = input
Remove mapping
Cancel
Save

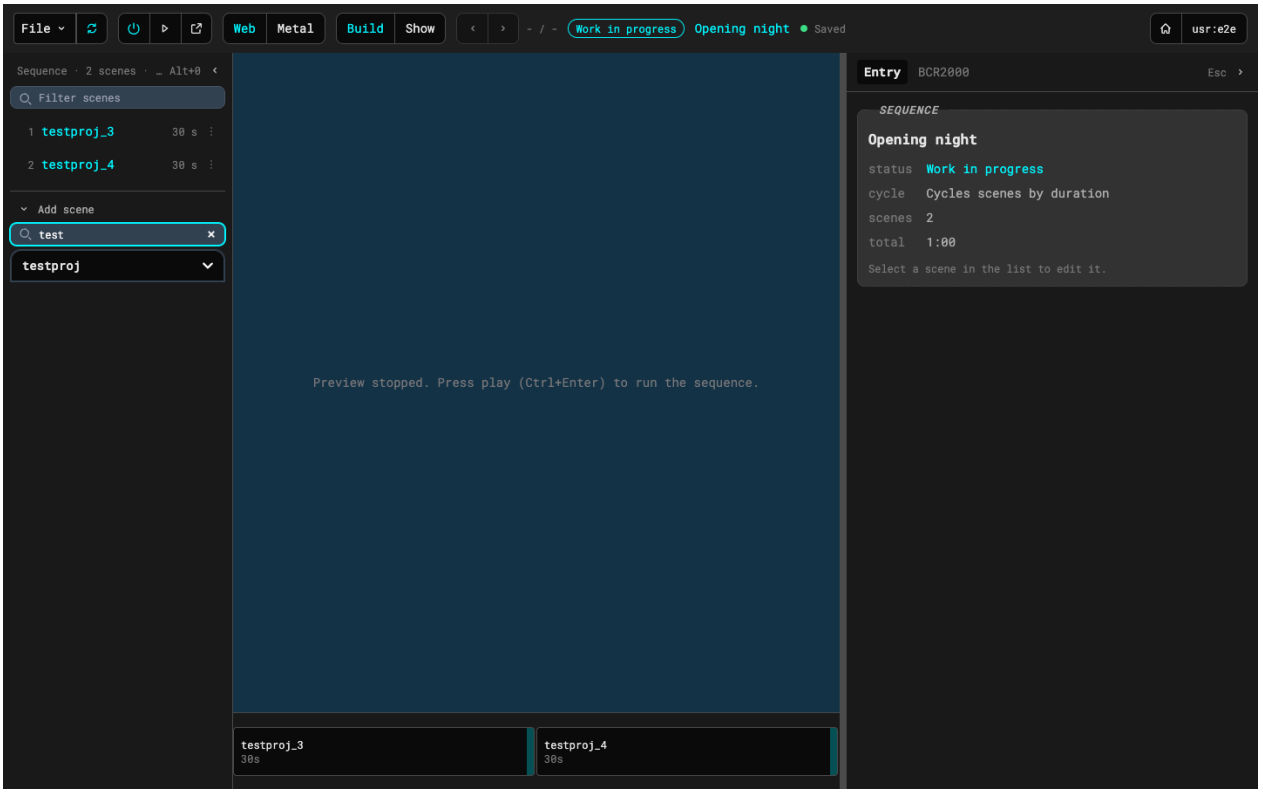
Sequences

A sequence is a list of scenes that play one after another. Open one from the Sequences tab on the home page. The page has the scene list on the left (Alt+0 hides it), the preview in the middle and a panel on the right with two tabs, Entry and BCR2000. The toolbar has the File menu, a cycle button, the preview buttons, a Build | Show switch, Previous and Next with the position (`1 / 2`), the status, the sequence's name and the save state. Save and Publish work like they do for a scene.

Building a sequence

Build is for putting the sequence together. Pick Add scene under the list, type part of a project's name, and choose a version to add it at the end. Each row shows the scene, its duration in seconds and a menu. Select a row to edit it in the Entry tab: choose another version of the scene, set its duration, move it up or down, or remove it. With nothing selected the Entry tab shows the sequence's status, whether it cycles, the number of scenes and the total time.



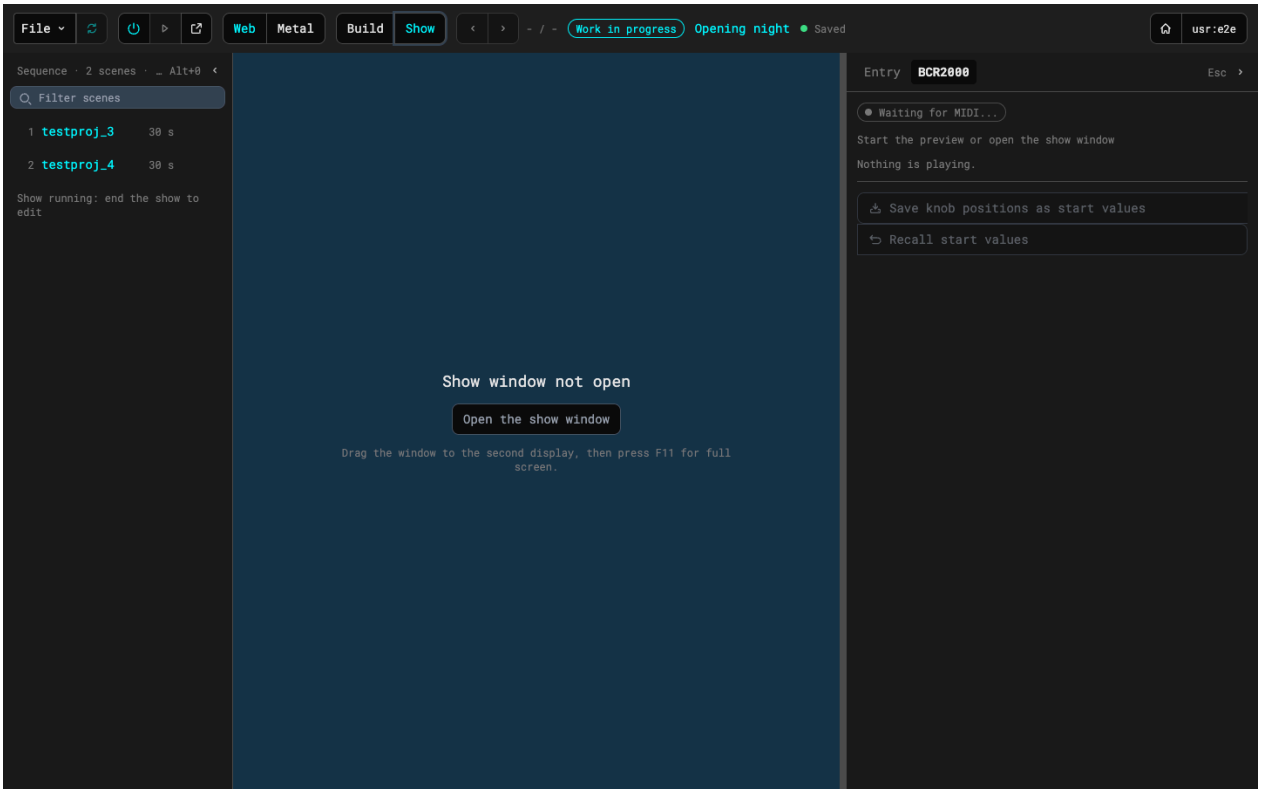


The cycle button (the two arrows) makes the sequence move on by itself when each scene's duration runs out, and start over at the end. Without it the sequence stays on a scene until you step. Play runs the sequence in the middle of the editor, and New tab runs it in its own browser tab. Both follow your edits.

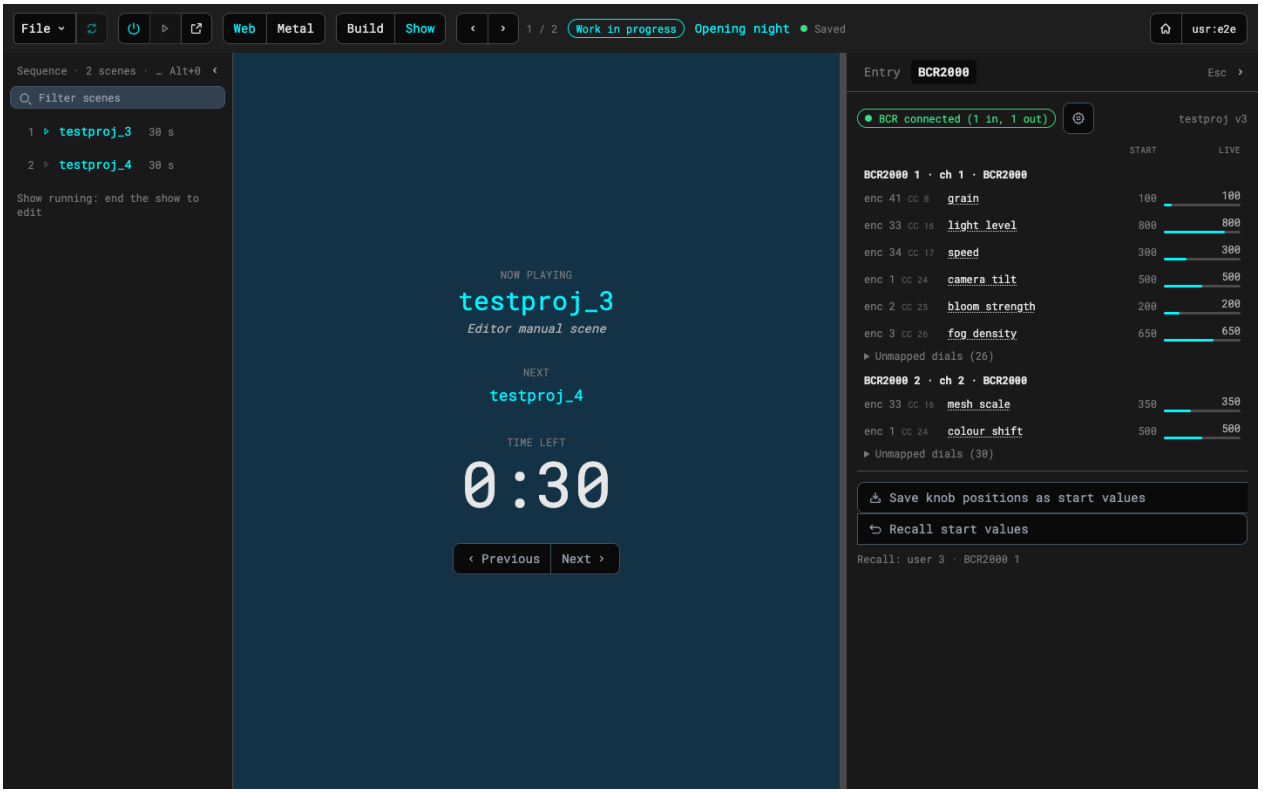
Showing a sequence

Show is for performing it. Switch to Show and the editor stops playing the preview itself: the show plays in its own window, so you can drag that window to the second display and press ^⌘F on a Mac (F11 on Windows) for full screen. If the window isn't open yet, the middle of the page says so, with a button that opens it. The show plays your unsaved copy of the sequence (the File menu says so), and the list can't be edited until you switch back to Build. You can still jump: hover over a row (or select it and press Enter) and press Jump to play that scene.

If you reload the editor while the show window is open, the page says Reconnecting… until it finds the window again. If you close the show window, the page says Show window closed, with the button to open it again.



When the window is open, the middle of the page shows what is playing now, what is next and the time left, with Previous and Next buttons. The toolbar's arrows, the ← and → keys, a BCR2000 button mapped to Previous or Next and the show window's own N and P keys do the same. If the sequence cycles, it moves on when the time runs out.



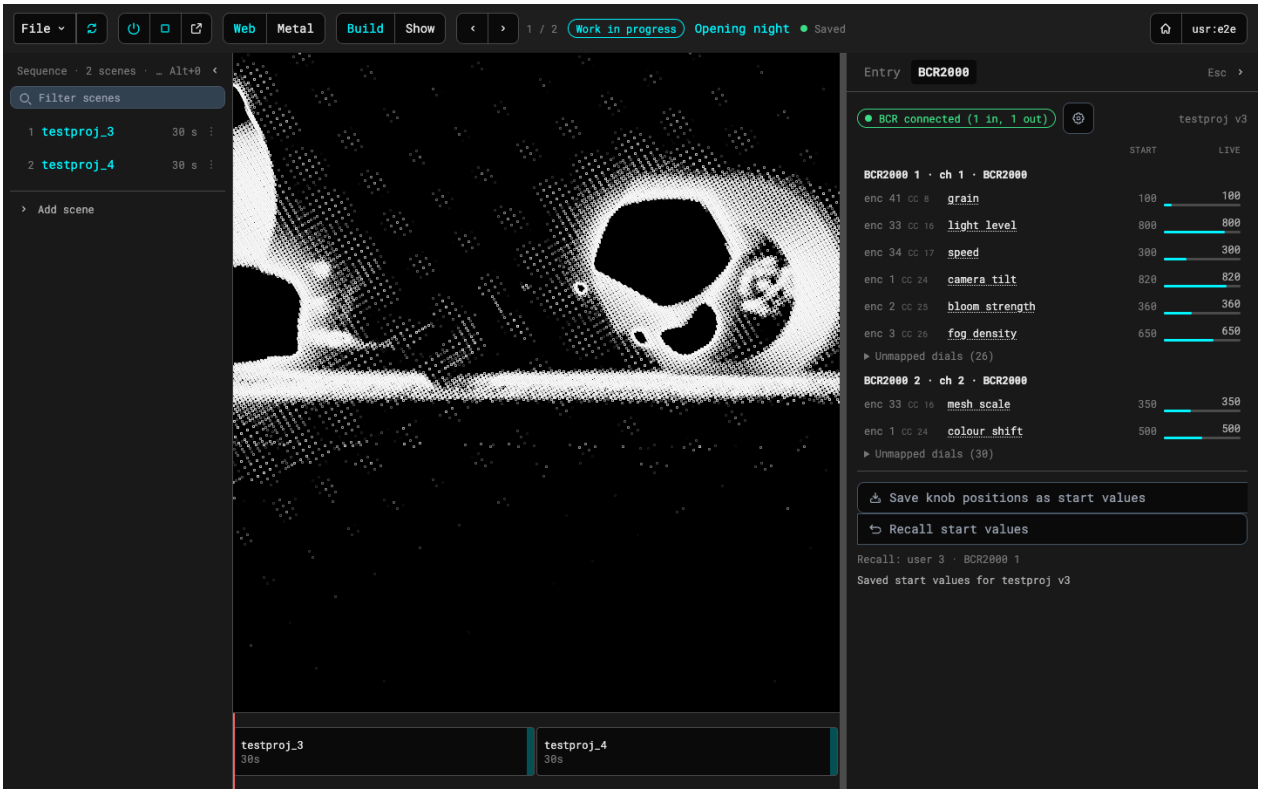
Without cycle the show is manual, and the layout changes. A strip under the toolbar shows NOW (the position, the scene and its description), NEXT with a Next → button, the Save and Recall messages, the BCR status chip, and the Save as start values and Recall start values buttons. The middle of the page is a grid for each controller of the playing scene: its knobs laid out like the hardware, with the push-encoder row on top, then the buttons, the three encoder rows, and the user and foot keys along the bottom. A knob shows its live value large in a ring (with a tick at the saved start value), its label and a small encoder number, and the Group 1 to 4 tabs in the grid's header follow the knobs as on the BCR2000 page. When a scene loads the dials already show the saved start values, as the unit's rings do, and a knob that's off its start value is highlighted. A lit dot on a button means it is held, or latched on for a toggle. Turn a knob or press a button on the hardware and its picture follows. Above the grids, Save as start values stores the knobs' positions as the scene's start values, and Recall start values sends them back to the knobs.



Build brings the preview back. The show window stays open until you close it.

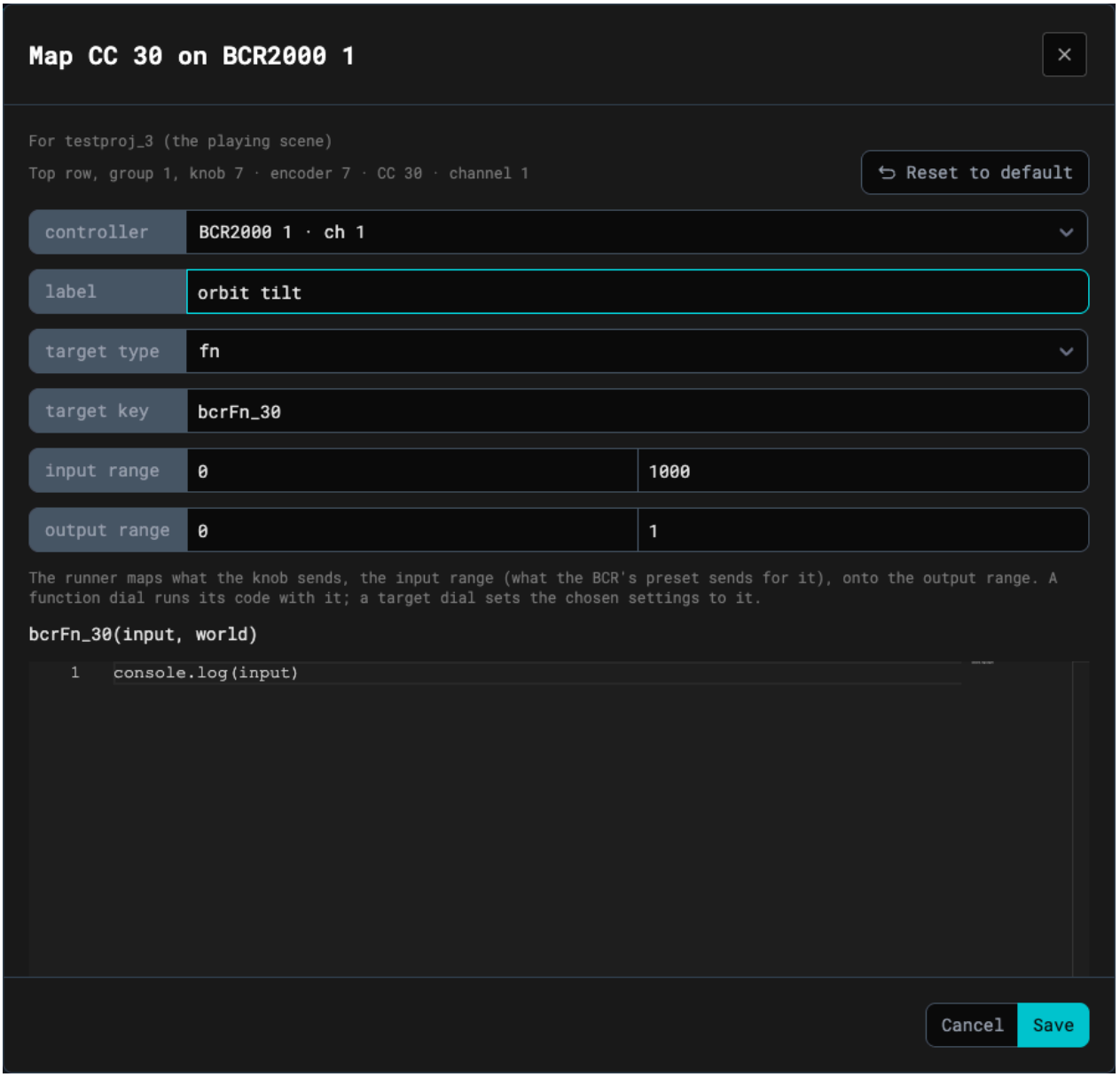
The BCR2000 tab

The BCR2000 tab lists the dials of whichever scene is playing, with their start values and the live values from the controller. A row whose knob is off its start value is highlighted, and its live value is coloured. The chip at the top says whether a BCR2000 is connected and how many MIDI inputs and outputs the browser sees. Each controller has its own heading, and dials that aren't mapped fold into an Unmapped dials list. The gear button beside the status chip opens the same menu as on the BCR2000 page, with MIDI ports, Controllers and Sync from BCR (no Remove BCR2000 mapping here); in Show the Controllers row is off, since a show can't change its controllers, and Sync from BCR still works. The tab follows the scene that is playing, not the row you selected; with nothing playing it has nothing to show. Clicking a dial opens the same dialog as on the BCR2000 page, including the Settings select of a target dial (see Target dials above). Save knob positions as start values (⌥⇧S on a Mac, Alt+Shift+S on Windows) saves the knobs' positions into the playing scene straight away, and Recall start values (⌥R, Alt+R) sends them back to the knobs.



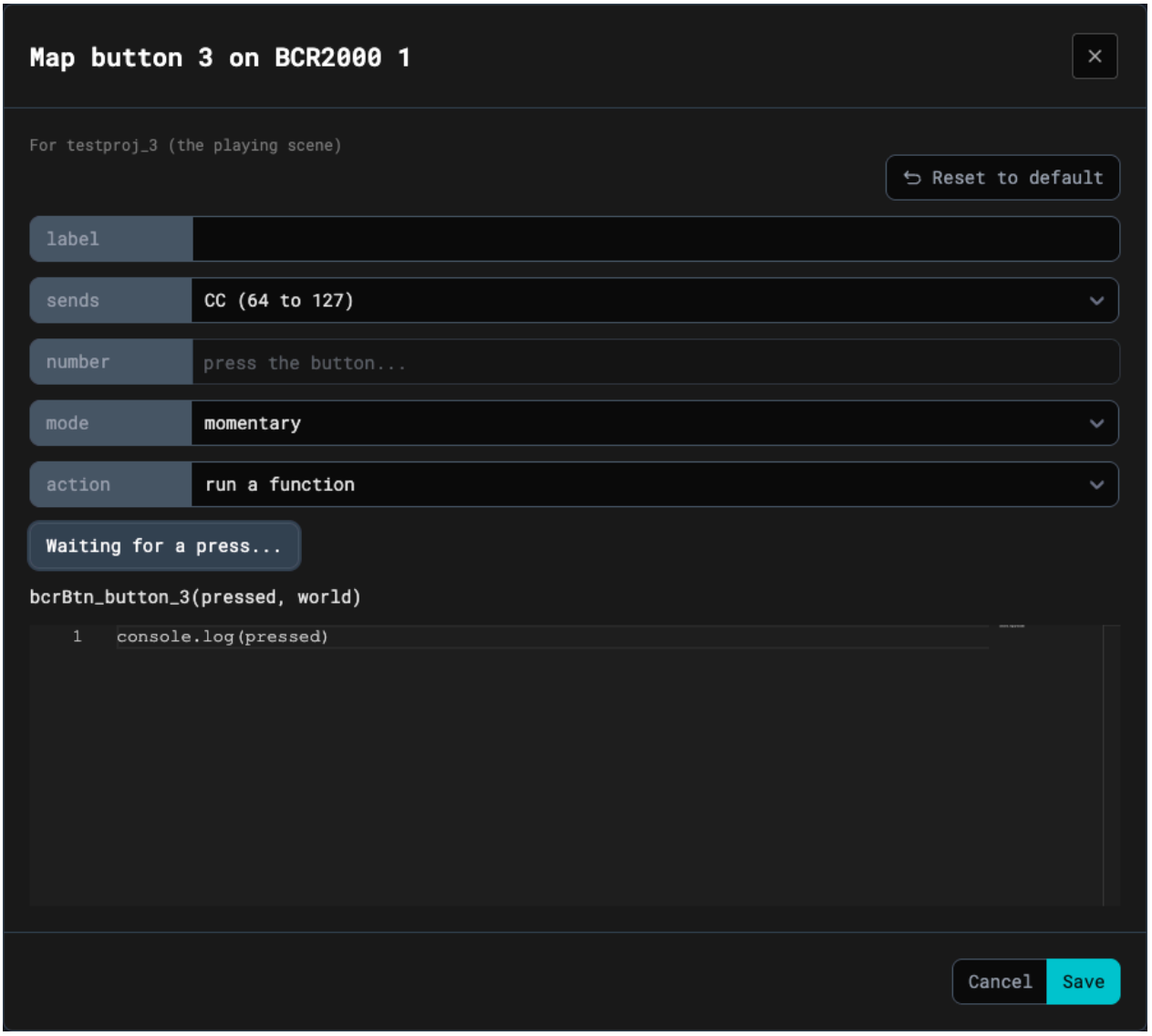
Click a mapped dial's name, or Map... on an unmapped one, to map it right here without leaving the sequence; in the manual show, click any knob or button in the

grid. The dialog is the one on the BCR2000 page, with the same fields: the controller, the label, what it controls, the input and output ranges, the NRPN number for groups 2 to 4 and a function, with Reset to default, Save and Cancel. Remove mapping drops a dial's mapping, with its function and start value. Every change is saved to the playing scene straight away, and the runner picks it up without restarting.

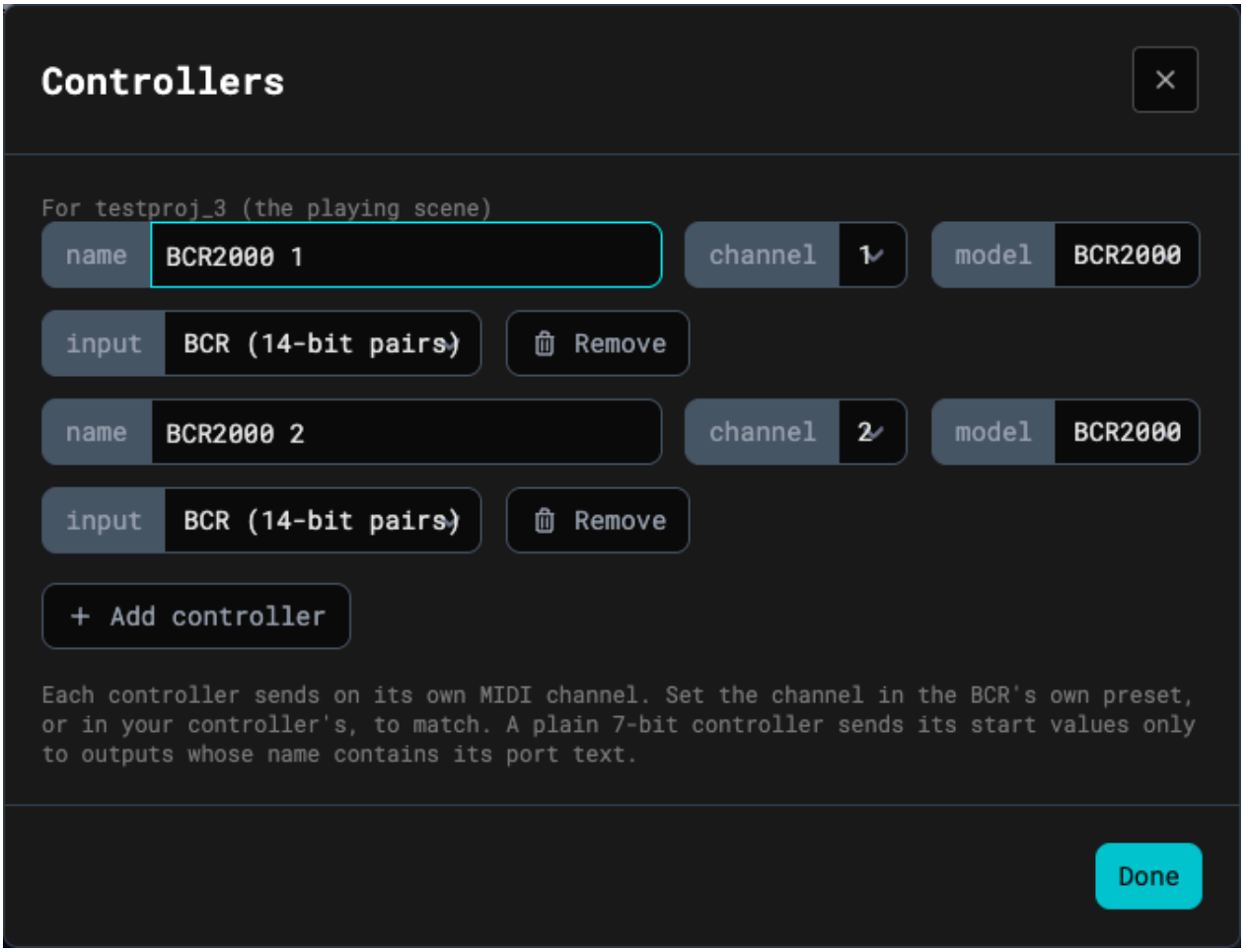


Buttons are mapped the same way. A button sends a CC (64 to 127) or a note. Press Learn and then the button on the controller to fill in its number. The mode is momentary or toggle, and its action runs a function, goes to the previous or next scene, or recalls the start values. Every scene starts with three buttons on the first controller: user 1 is Previous scene, user 2 is Next scene and user 3 is Recall start

values (T1 to T3 on a BCR32). No button saves start values, so a show can't overwrite them by a stray press; use the button or the keys.



The gear menu's Controllers row changes a controller's name, its MIDI channel and its model (BCR2000 or BCR32), adds another controller on its own channel, and can remove one; these changes are saved to the playing scene straight away.

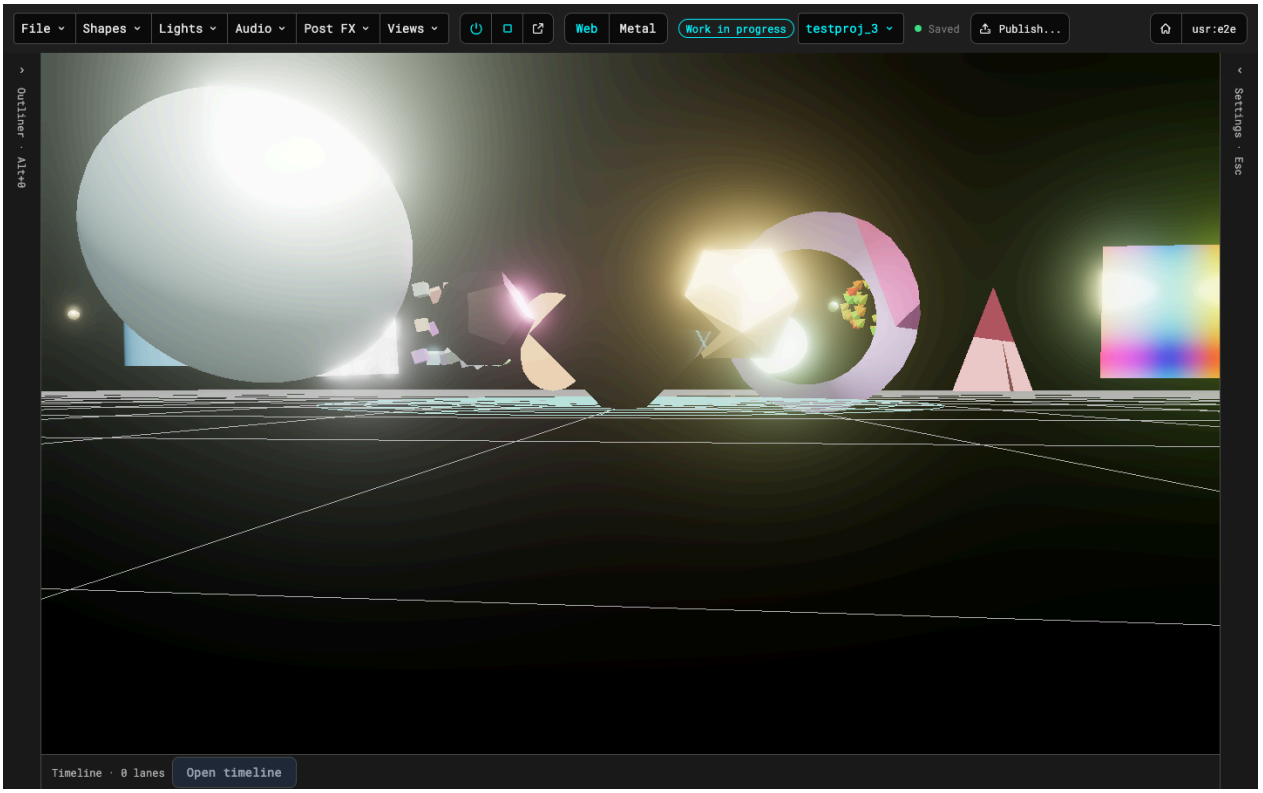

Controllers
For testproj_3 (the playing scene)
name
BCR2000 1
channel
1
model
BCR2000
input
BCR (14-bit pairs)
Remove
name
BCR2000 2
channel
2
model
BCR2000
input
BCR (14-bit pairs)
Remove
+ Add controller
Each controller sends on its own MIDI channel. Set the channel in the BCR's own preset, or in your controller's, to match. A plain 7-bit controller sends its start values only to outputs whose name contains its port text.
Done

Preview

The three buttons next to the Views menu run the preview.

- **Power** (start GL Engine) starts the runner, the program that draws the scene. It lights up while the runner is on. Start it first.
- **Play** shows the open version in the middle of the editor, including edits you haven't saved. It turns into a stop button while playing.
- **Web | Metal** chooses the runner that Play's new-tab button and the sequence page's Show window open: the web runner in a browser window, or the Metal runner, the native Mac app. With Metal chosen, Power starts and stops the Metal runner. The preview in the middle of the editor is always the web runner.
- **New tab** (the button's name, "preview in new tab") shows the same preview in a window of its own. Clicking it again brings that window back to the front instead of opening another. If you reload the editor, the window keeps playing and the editor finds it again within a few seconds, so Recall start values and your edits reach it.

While a preview is open (in the editor or in its own window), it follows your edits. About half a second after you stop changing something, the scene rebuilds and the camera stays where you left it.



Collapse the outliner and the inspector to give the preview the whole window; Esc shows and hides the inspector, also while the preview has focus.

Keys in the preview itself: H shows or hides the runner's info panel (it starts hidden; it lists what the scene uses that the runner can't draw yet, and any warnings), and N and P step to the next and previous scene when the preview is playing a sequence. On the sequence page, → and ← step too, from the page or the show window.

Saving, versions and status

A project holds versions of a scene. Each version is saved on its own and has its own description, status and sharing.

- **Save** (File > Save) writes the open version to the database. The toolbar says Saved when everything is saved and Unsaved when it isn't.
- **New version** adds an empty version to the project and opens it.
- **Duplicate version** adds a copy of the open version, edits included, and opens it.
- **Save version to new project…** asks for a title and copies the open version into a new project.
- **Save version to other project…** copies the open version into a project you choose.
- **Edit description…** changes the text shown for the version in the version menu and the project list.
- **Share version…** lists the other users; tick the ones who should see the version.
- **Delete version…** removes the open version after you confirm. A project's last version can't be deleted here; delete the project from the Projects page.

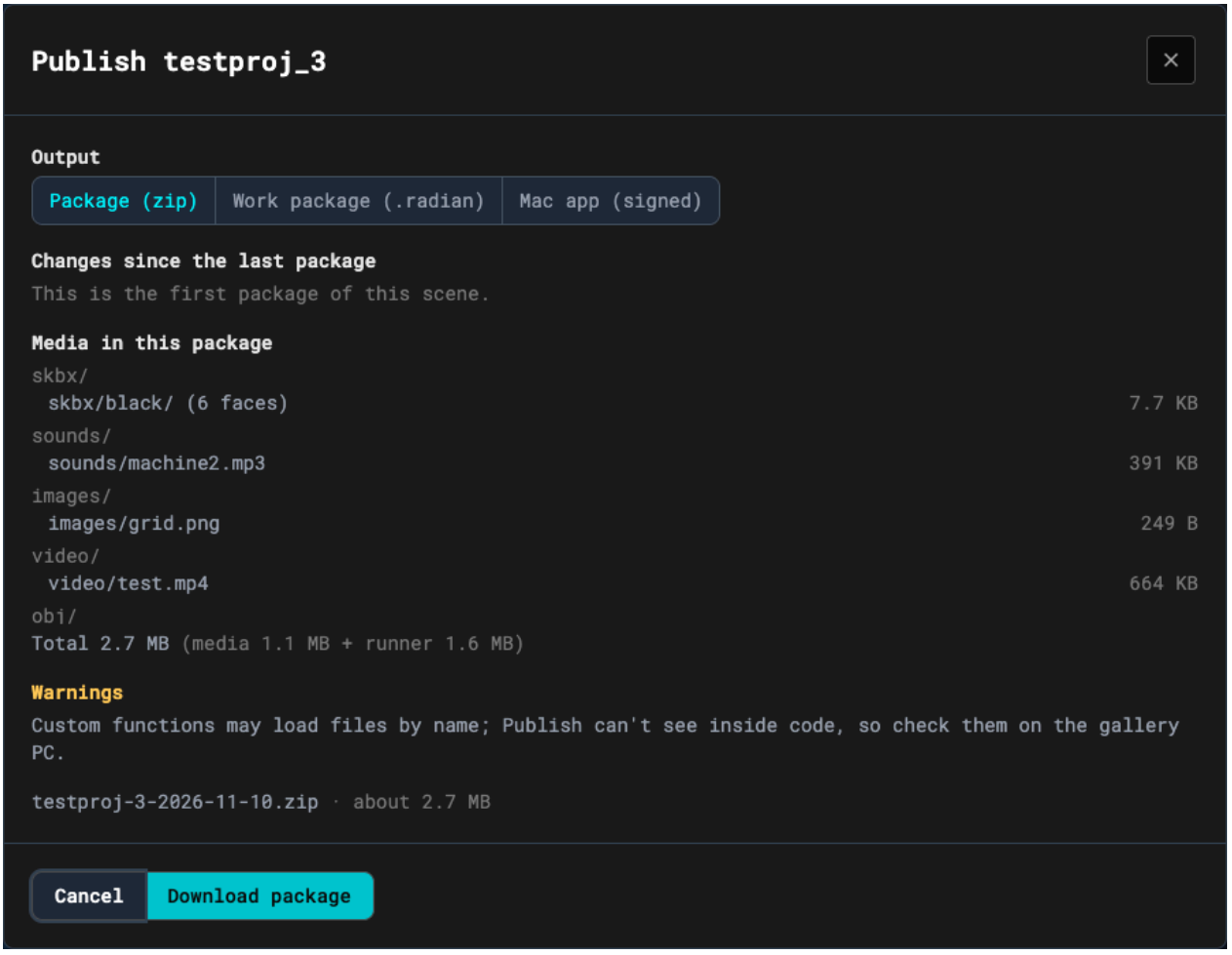
To change the version you're editing, pick it in the version menu. The status pill (Status) marks where a version stands: Work in progress, Review, Fave or Complete.

Publish

Publish makes a package that plays a scene on another computer (a gallery machine) with only a browser: no editor, network or database. It leaves the shared media folder alone.

Open File > Publish... (or the Publish... button on the toolbar). The dialog is named for the version, for example Publish testproj_3. It shows:

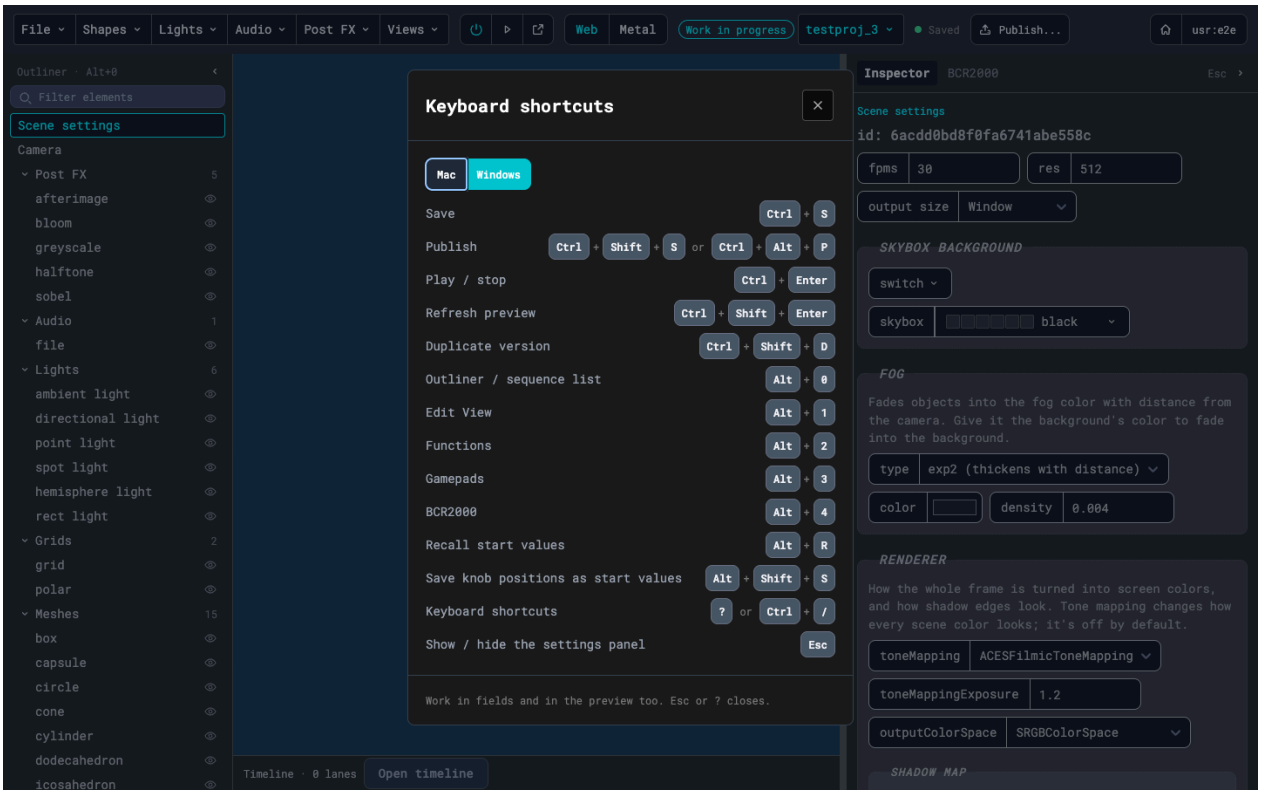
- Changes since the last package, or that this is the first package of the scene.
- Media in this package, each file with its size, and the package's total size.
- Fix before publishing: problems that stop the package, such as a file the scene names that isn't in the media folder. Fix them in the editor and open Publish again.
- Warnings, such as custom functions that may load files by name; Publish can't see inside code, so check them on the gallery machine.



Press Download package. It builds a zip named for the project, version and date, and your browser downloads it. Copy the zip to the gallery machine. Unzipping it, starting it, and making it start with the PC are in the package's own instructions: `apps/runner/package/README.txt` in the repo.

Keyboard shortcuts

Press `?` (or ⌘/ on a Mac, Ctrl+/ on Windows) to see the list in the editor. The Alt number keys, ⌥R, ⌥⇧S and `?` don't fire while you're typing in a field, and ⌘/ or Ctrl+/ leaves the code editor's own comment key alone. The arrows that step a sequence don't fire on a slider, a select or a list, or in a dialog. Save, Publish, Play, Refresh and Duplicate version work everywhere.



Action	Mac	Windows
Save	⌘S	Ctrl+S
Publish	⇧⌘S	Ctrl+Shift+S
Publish	⌥⌘P	Ctrl+Alt+P
Play / stop	⌘↵	Ctrl+Enter
Refresh preview	⇧⌘↵	Ctrl+Shift+Enter
Duplicate version	⇧⌘D	Ctrl+Shift+D
Outliner / sequence list	⌥0	Alt+0
Edit View	⌥1	Alt+1
Functions	⌥2	Alt+2
Gamepads	⌥3	Alt+3
BCR2000	⌥4	Alt+4
Recall start values	⌥R	Alt+R
Save knob positions as start values	⌥⇧S	Alt+Shift+S
Next scene	→	→
Previous scene	←	←
Keyboard shortcuts	?	?
Keyboard shortcuts	⌘/	Ctrl+/
Show / hide the settings panel	Esc	Esc

The runner's own keys, when the preview has focus:

Key	What it does
H	Shows or hides the runner's info panel
N	Next scene in a sequence
P	Previous scene in a sequence