Contents

Nexus Ultimate Prop Controller API Reference v1.1

This guide describes the available HTTP control and configuration endpoints.

Control Quickstart

Use this sequence for reliable controller automation:

1. `GET /api/status` and `GET /api/state?minimal=1` to discover readiness and current state.
2. Optionally read config sections (`/api/io`, `/api/logic`, `/api/lighting`, `/api/integrations`) before planning changes.
3. Write config sections with `POST` to section endpoints (`/api/logic`, `/api/io`, etc.) using complete JSON documents.
4. Run the prop through normal gameplay flow and operator interactions.
5. Verify runtime behavior with `GET /api/events?since=<cursor>&max=<n>` and `GET /api/inputs`.

How Things Interact

- **States** are the high-level lifecycle (for example: Ready, Armed, Solved, Resetting). Use `set_state` to move between phases.
- **Logic Rules** evaluate conditions and fire when matched. They are best used as decision points, then linked to actions/sequences.
- **Action Triggers** are reusable output bundles. Use `action_trigger` when you want one named behavior to be callable from multiple places.
- **Operator Controls** are manual commands used by operators during runtime. Use `operator_control` for supervised overrides.
- **Input Sequences** (ordered/timed) detect player patterns. When complete, they typically drive state changes or triggers.
- **Output Sequences** run timed multi-step effects and are useful for coordinated light/sound/mechanical choreography.
- **Live Monitor / Events** provide runtime observability. Pair action calls with `/api/events` and `/api/inputs` to validate behavior.
- **Integrations** connect the controller to external systems. In the current UI surface, external control focuses on MQTT/HTTP/OSC.

Configuration-First Runtime Examples

Example 1: Standard Puzzle Lifecycle

- Configure states in logic design as `Ready -> Armed -> Solved -> Resetting -> Ready`.
- Use `POST /api/logic` to save rules that move between those states from input conditions.
- Use `POST /api/operator_controls` to define explicit operator reset/rearm controls.
- Runtime expectation: normal player flow advances automatically; operator can recover cleanly to `Ready`.

Example 2: Ordered Input Sequence Unlock

- Configure the sequence in logic/input sequence sections (saved via `POST /api/logic`).
- Configure output effects (lights/sound/actuator) via `POST /api/lighting`, `POST /api/lighting_sequences`, and relevant output config endpoints.
- Runtime expectation: correct sequence triggers unlock behavior; incorrect progress follows configured reset/penalty behavior.

Example 3: Multi-Output Choreography on Solve

- Configure a reusable output sequence for the solve moment (lights + sound + motion timing).
- Save output sequence/runtime links in `POST /api/logic` and `POST /api/lighting_sequences`.
- Runtime expectation: a single solve condition triggers a deterministic timed show without manual intervention.

Example 4: Recovery and Safety Behavior

- Configure fallback/reset paths in logic rules and operator controls.
- Persist connection/identity and safety defaults using `POST /api/save_connection` and relevant config endpoints.
- Runtime expectation: after fault, reboot, or aborted attempt, operators can reliably return the prop to the known safe starting state.

Example 5: Verification During Commissioning

- After saving configuration, verify status with `GET /api/status` and `GET /api/state?minimal=1`.
- Observe live behavior with `GET /api/events?since=<cursor>&max=<n>` and `GET /api/inputs`.
- Runtime expectation: event stream matches intended sequence of states, triggers, and outputs.

HTTP Endpoint Index

Auth marks: `-` none, `S` simple auth, `A` admin auth, `A/S` either.

Method	Path	Auth
POST	/api/action	-
POST	/api/action_prewarm	A/S
GET	/api/action_prewarm/status	A/S
GET	/api/auth/logout	-
POST	/api/auth/logout	-
POST	/api/cmd	-
GET	/api/config	-
POST	/api/config	A
GET	/api/config_snapshot	-
GET	/api/connection_config	A/S
GET	/api/device/identify	A/S
POST	/api/device/identify	-
POST	/api/device/identify	A/S
POST	/api/device/neuralyze	-
POST	/api/device/reboot	A
GET	/api/diagnostics/io	-
GET	/api/dmx	-
POST	/api/dmx	A/S
GET	/api/dmx/live	A/S
GET	/api/events	-
GET	/api/events/export	A/S
GET	/api/events/persistent/export	A/S
POST	/api/events/send	A/S
POST	/api/factory_reset	A
POST	/api/factory_reset_keep_network	A
GET	/api/fs_test	-
POST	/api/i2c_scan	A/S
GET	/api/inbound	-
POST	/api/inbound	-
GET	/api/inputs	-
GET	/api/integrations	A/S
POST	/api/integrations	A
GET	/api/io	A/S
POST	/api/io	A/S
GET	/api/io/diagnostics	-
POST	/api/io/test	-
GET	/api/lighting	-
POST	/api/lighting	A/S
GET	/api/lighting/status	-
GET	/api/lighting_effects	-
POST	/api/lighting_effects	A/S

Method	Path	Auth
GET	/api/lighting_palettes	-
POST	/api/lighting_palettes	A/S
GET	/api/lighting_sequences	-
POST	/api/lighting_sequences	A/S
GET	/api/logic	-
POST	/api/logic	A/S
GET	/api/logic_sections	-
POST	/api/logic_sections	A/S
POST	/api/logic_validate	A
GET	/api/operator_config	-
GET	/api/operator_controls	-
POST	/api/operator_controls	A
POST	/api/ota/apply	A
GET	/api/ota/check	A
POST	/api/ota/upload	A
GET	/api/ping	-
GET	/api/ping	-
POST	/api/reboot	A
POST	/api/recovery_mode	A
GET	/api/rfid/status	A/S
POST	/api/rfid/write	A
GET	/api/runtime_debug	A/S
POST	/api/save_connection	A/S
POST	/api/setup_abort_wizard	-
POST	/api/setup_abort_wizard	-
POST	/api/setup_apply_preset	A
POST	/api/setup_apply_preset	A
POST	/api/setup_complete	A
POST	/api/setup_complete	A
POST	/api/setup_finalize	-
POST	/api/setup_finalize	-
GET	/api/setup_presets	-
GET	/api/setup_presets	-
GET	/api/setup_status	-
GET	/api/setup_status	-
GET	/api/simple_config	A/S
GET	/api/simple_ui	-
POST	/api/simple_ui	A/S
GET	/api/sound/debug	A/S
GET	/api/sound/files	A/S
POST	/api/sound/files	A/S
GET	/api/sound/tracks	A/S
POST	/api/sound/upload	A/S
GET	/api/state	-
GET	/api/status	-
POST	/api/test/io	-
POST	/api/wifi_connect	-
POST	/api/wifi_connect	A/S
GET	/api/wifi_connect_status	-
GET	/api/wifi_connect_status	A/S
GET	/api/wifi_scan	-
GET	/api/wifi_scan	-
POST	/api/wizard_apply	-
POST	/api/wizard_apply	-

Endpoint Details

`POST /api/action`

- Auth: -
- Params: plain
- Body Keys: action, payload, suppress_log, target
- Notes: returns 400 when request body is missing; returns 400 on invalid JSON; requires action in JSON body

`POST /api/action_prewarm`

- Auth: A/S
- Params: plain
- Body Keys: append, max_bytes, max_plans
- Notes: returns 403 when unauthorized

`GET /api/action_prewarm/status`

- Auth: A/S
- Params: none
- Body Keys: none
- Notes: returns 403 when unauthorized

`GET /api/auth/logout`

- Auth: –
- Params: none
- Body Keys: none
- Notes: none

`POST /api/auth/logout`

- Auth: –
- Params: none
- Body Keys: none
- Notes: none

`POST /api/cmd`

- Auth: –
- Params: plain
- Body Keys: none
- Notes: returns 400 when request body is missing

`GET /api/config`

- Auth: –
- Params: none
- Body Keys: none
- Notes: none

`POST /api/config`

- Auth: A
- Params: plain
- Body Keys: integrations, io, logic, operator_controls
- Notes: returns 400 when request body is missing; returns 400 on invalid JSON; returns 403 when unauthorized

`GET /api/config_snapshot`

- Auth: –
- Params: none

- Body Keys: none
- Notes: none

`GET /api/connection_config`

- Auth: `A/S`
- Params: none
- Body Keys: none
- Notes: returns 403 when unauthorized

`GET /api/device/identify`

- Auth: `A/S`
- Params: none
- Body Keys: none
- Notes: returns 403 when unauthorized

`POST /api/device/identify`

- Auth: `-`
- Params: plain
- Body Keys: enabled
- Notes: none

`POST /api/device/identify`

- Auth: `A/S`
- Params: plain
- Body Keys: enabled
- Notes: returns 403 when unauthorized

`POST /api/device/neuralyze`

- Auth: `-`
- Params: plain
- Body Keys: confirm, reset
- Notes: returns 400 when request body is missing; returns 400 on invalid JSON

`POST /api/device/reboot`

- Auth: `A`
- Params: none
- Body Keys: none
- Notes: returns 403 when unauthorized

`GET /api/diagnostics/io`

- Auth: `-`
- Params: none
- Body Keys: none
- Notes: none

GET /api/dmx

- Auth: –
- Params: none
- Body Keys: none
- Notes: none

POST /api/dmx

- Auth: A/S
- Params: allow_empty, persist, plain
- Body Keys: none
- Notes: returns 400 when request body is missing; returns 403 when unauthorized

GET /api/dmx/live

- Auth: A/S
- Params: count, start
- Body Keys: none
- Notes: returns 403 when unauthorized

GET /api/events

- Auth: –
- Params: max, since
- Body Keys: none
- Notes: none

GET /api/events/export

- Auth: A/S
- Params: none
- Body Keys: none
- Notes: returns 403 when unauthorized

GET /api/events/persistent/export

- Auth: A/S
- Params: none
- Body Keys: none
- Notes: returns 403 when unauthorized

POST /api/events/send

- Auth: A/S
- Params: none
- Body Keys: none
- Notes: returns 403 when unauthorized

POST /api/factory_reset

- Auth: A
- Params: plain

- Body Keys: confirm
- Notes: returns 400 when request body is missing; returns 400 on invalid JSON; returns 403 when unauthorized

`POST /api/factory_reset_keep_network`

- Auth: `A`
- Params: plain
- Body Keys: confirm, prop_wizard, setup_mode
- Notes: returns 400 when request body is missing; returns 400 on invalid JSON; returns 403 when unauthorized

`GET /api/fs_test`

- Auth: `-`
- Params: none
- Body Keys: none
- Notes: none

`POST /api/i2c_scan`

- Auth: `A/S`
- Params: plain
- Body Keys: io, io_json, scl, sda
- Notes: returns 403 when unauthorized

`GET /api/inbound`

- Auth: `-`
- Params: path, payload
- Body Keys: none
- Notes: requires path (for inbound route-form requests)

`POST /api/inbound`

- Auth: `-`
- Params: path, payload, plain
- Body Keys: action, path, payload
- Notes: returns 400 on invalid JSON; requires path (for inbound route-form requests)

`GET /api/inputs`

- Auth: `-`
- Params: none
- Body Keys: none
- Notes: none

`GET /api/integrations`

- Auth: `A/S`
- Params: none
- Body Keys: none
- Notes: returns 403 when unauthorized

POST /api/integrations

- Auth: A
- Params: allow_empty, persist, plain, sync
- Body Keys: none
- Notes: returns 400 when request body is missing; returns 400 on invalid JSON; returns 403 when unauthorized

GET /api/io

- Auth: A/S
- Params: none
- Body Keys: none
- Notes: returns 403 when unauthorized

POST /api/io

- Auth: A/S
- Params: allow_empty, plain, sync
- Body Keys: none
- Notes: returns 400 when request body is missing; returns 403 when unauthorized

GET /api/io/diagnostics

- Auth: -
- Params: none
- Body Keys: none
- Notes: none

POST /api/io/test

- Auth: -
- Params: none
- Body Keys: none
- Notes: none

GET /api/lighting

- Auth: -
- Params: none
- Body Keys: none
- Notes: none

POST /api/lighting

- Auth: A/S
- Params: allow_destructive, allow_empty, persist, plain, sync
- Body Keys: none
- Notes: returns 400 when request body is missing; returns 403 when unauthorized

GET /api/lighting/status

- Auth: -
- Params: none

- Body Keys: none
- Notes: none

GET /api/lighting_effects

- Auth: –
- Params: none
- Body Keys: none
- Notes: none

POST /api/lighting_effects

- Auth: A/S
- Params: plain
- Body Keys: none
- Notes: returns 400 when request body is missing; returns 403 when unauthorized

GET /api/lighting_palettes

- Auth: –
- Params: none
- Body Keys: none
- Notes: none

POST /api/lighting_palettes

- Auth: A/S
- Params: plain
- Body Keys: none
- Notes: returns 400 when request body is missing; returns 403 when unauthorized

GET /api/lighting_sequences

- Auth: –
- Params: none
- Body Keys: none
- Notes: none

POST /api/lighting_sequences

- Auth: A/S
- Params: allow_empty, plain
- Body Keys: none
- Notes: returns 400 when request body is missing; returns 403 when unauthorized

GET /api/logic

- Auth: –
- Params: none
- Body Keys: none
- Notes: none

`POST /api/logic`

- Auth: `A/S`
- Params: plain
- Body Keys: none
- Notes: returns 400 when request body is missing; returns 403 when unauthorized

`GET /api/logic_sections`

- Auth: `-`
- Params: keys
- Body Keys: none
- Notes: none

`POST /api/logic_sections`

- Auth: `A/S`
- Params: plain
- Body Keys: none
- Notes: returns 400 when request body is missing; returns 403 when unauthorized

`POST /api/logic_validate`

- Auth: `A`
- Params: plain
- Body Keys: none
- Notes: returns 400 when request body is missing; returns 403 when unauthorized

`GET /api/operator_config`

- Auth: `-`
- Params: none
- Body Keys: none
- Notes: none

`GET /api/operator_controls`

- Auth: `-`
- Params: none
- Body Keys: none
- Notes: none

`POST /api/operator_controls`

- Auth: `A`
- Params: allow_empty, plain, sync
- Body Keys: none
- Notes: returns 400 when request body is missing; returns 403 when unauthorized

`POST /api/ota/apply`

- Auth: `A`
- Params: plain

- Body Keys: override, url
- Notes: returns 403 when unauthorized

GET /api/ota/check

- Auth: A
- Params: none
- Body Keys: none
- Notes: returns 403 when unauthorized

POST /api/ota/upload

- Auth: A
- Params: none
- Body Keys: none
- Notes: returns 403 when unauthorized

GET /api/ping

- Auth: -
- Params: none
- Body Keys: none
- Notes: none

GET /api/ping

- Auth: -
- Params: none
- Body Keys: none
- Notes: none

POST /api/reboot

- Auth: A
- Params: none
- Body Keys: none
- Notes: returns 403 when unauthorized

POST /api/recovery_mode

- Auth: A
- Params: none
- Body Keys: none
- Notes: returns 403 when unauthorized

GET /api/rfid/status

- Auth: A/S
- Params: none
- Body Keys: none
- Notes: returns 403 when unauthorized

POST /api/rfid/write

- Auth: A
- Params: plain
- Body Keys: block, id, value
- Notes: returns 400 when request body is missing; returns 400 on invalid JSON; returns 403 when unauthorized

GET /api/runtime_debug

- Auth: A/S
- Params: none
- Body Keys: none
- Notes: returns 403 when unauthorized

POST /api/save_connection

- Auth: A/S
- Params: plain
- Body Keys: ap_hidden, device_name, eth_dhcp, eth_dns1, eth_dns2, eth_gateway, eth_ip, eth_subnet, mqtt_host, mqtt_ip, mqtt_user, room_prefix, wifi_dhcp, wifi_dns1, wifi_dns2, wifi_gateway, wifi_ip, wifi_ssid, wifi_ssid_clear, wifi_subnet
- Notes: returns 400 when request body is missing; returns 400 on invalid JSON; returns 403 when unauthorized

POST /api/setup_abort_wizard

- Auth: -
- Params: none
- Body Keys: none
- Notes: none

POST /api/setup_abort_wizard

- Auth: -
- Params: none
- Body Keys: none
- Notes: none

POST /api/setup_apply_preset

- Auth: A
- Params: plain
- Body Keys: dmx, integrations, io, lighting, lighting_sequences, logic, operator_controls, operator_views, simple_ui
- Notes: returns 400 when request body is missing; returns 400 on invalid JSON; returns 403 when unauthorized

POST /api/setup_apply_preset

- Auth: A
- Params: plain
- Body Keys: dmx, integrations, io, lighting, lighting_sequences, logic, operator_controls, operator_views, simple_ui
- Notes: returns 400 when request body is missing; returns 400 on invalid JSON; returns 403 when unauthorized

POST /api/setup_complete

- Auth: A

- Params: plain
- Body Keys: ap_hidden, device_name, mode, set_wifi, try_wifi_connect, wifi_ssid
- Notes: returns 400 when request body is missing; returns 400 on invalid JSON; returns 403 when unauthorized

POST /api/setup_complete

- Auth: A
- Params: plain
- Body Keys: ap_hidden, device_name, mode, mqtt_host, mqtt_ip, mqtt_user, set_admin, set_wifi, try_wifi_connect, wifi_ssid
- Notes: returns 400 when request body is missing; returns 400 on invalid JSON; returns 403 when unauthorized

POST /api/setup_finalize

- Auth: –
- Params: none
- Body Keys: none
- Notes: none

POST /api/setup_finalize

- Auth: –
- Params: none
- Body Keys: none
- Notes: none

GET /api/setup_presets

- Auth: –
- Params: none
- Body Keys: none
- Notes: none

GET /api/setup_presets

- Auth: –
- Params: none
- Body Keys: none
- Notes: none

GET /api/setup_status

- Auth: –
- Params: none
- Body Keys: none
- Notes: none

GET /api/setup_status

- Auth: –
- Params: none
- Body Keys: none
- Notes: none

GET /api/simple_config

- Auth: A/S
- Params: none
- Body Keys: none
- Notes: returns 403 when unauthorized

GET /api/simple_ui

- Auth: –
- Params: none
- Body Keys: none
- Notes: none

POST /api/simple_ui

- Auth: A/S
- Params: plain
- Body Keys: controls, enabled, level, operator_views
- Notes: returns 400 when request body is missing; returns 400 on invalid JSON; returns 403 when unauthorized

GET /api/sound/debug

- Auth: A/S
- Params: id
- Body Keys: none
- Notes: returns 403 when unauthorized

GET /api/sound/files

- Auth: A/S
- Params: id
- Body Keys: none
- Notes: returns 403 when unauthorized

POST /api/sound/files

- Auth: A/S
- Params: plain
- Body Keys: none
- Notes: returns 403 when unauthorized

GET /api/sound/tracks

- Auth: A/S
- Params: id
- Body Keys: none
- Notes: returns 403 when unauthorized

POST /api/sound/upload

- Auth: A/S
- Params: none

- Body Keys: none
- Notes: returns 403 when unauthorized

`GET /api/state`

- Auth: –
- Params: minimal
- Body Keys: none
- Notes: none

`GET /api/status`

- Auth: –
- Params: none
- Body Keys: none
- Notes: none

`POST /api/test/io`

- Auth: –
- Params: none
- Body Keys: none
- Notes: none

`POST /api/wifi_connect`

- Auth: –
- Params: ssid
- Body Keys: none
- Notes: returns 409 if a Wi-Fi connect request is already running; returns 409 on duplicate connect

`POST /api/wifi_connect`

- Auth: A/S
- Params: ssid
- Body Keys: none
- Notes: returns 409 if a Wi-Fi connect request is already running; returns 409 on duplicate connect; returns 403 when unauthorized

`GET /api/wifi_connect_status`

- Auth: –
- Params: none
- Body Keys: none
- Notes: none

`GET /api/wifi_connect_status`

- Auth: A/S
- Params: none
- Body Keys: none
- Notes: returns 403 when unauthorized

`GET /api/wifi_scan`

- Auth: –
- Params: none
- Body Keys: none
- Notes: none

`GET /api/wifi_scan`

- Auth: –
- Params: none
- Body Keys: none
- Notes: none

`POST /api/wizard_apply`

- Auth: –
- Params: none
- Body Keys: none
- Notes: none

`POST /api/wizard_apply`

- Auth: –
- Params: none
- Body Keys: none
- Notes: none

Config Section Payload Guide

These endpoints expect complete JSON section payloads (not partial patch operations):

Endpoint	Section	Notes
`POST /api/io`	I/O config	Use full I/O JSON document.
`POST /api/logic`	Logic config	Use full logic JSON; can use `allow_destructive`/`allow_empty` query args.
`POST /api/lighting`	Lighting config	Use full lighting JSON.
`POST /api/lighting_sequences`	Lighting sequence config	Use full sequence JSON.
`POST /api/dmx`	DMX config	Use full DMX JSON.
`POST /api/integrations`	Integrations config	Use full integrations JSON.
`POST /api/operator_controls`	Operator controls config	Use full operator controls JSON.
`POST /api/config`	Multi-section batch update	Combined update for multiple sections.

`/api/action` Command Catalog

All commands below are accepted via:

```
POST /api/action
Content-Type: application/json

{"action":"<command>","target":"<string>","payload":"<string-or-json-string>","suppress_log":false}
```

`action`	Purpose
`set_var`	Set a runtime variable value.
`var_inc`	Increment a runtime variable.
`var_dec`	Decrement a runtime variable.

action	Purpose
var_setpoint	Setpoint-style variable update.
random_var	Set a runtime variable to a random number or a random value from a supplied set.
variable_function	Wrapper for variable helpers such as random, increment/decrement, and fade.
var_fade_start	Fade a numeric runtime variable toward a target value over time.
var_fade_pause	Pause an active variable fade.
var_fade_clear	Clear an active variable fade.
timer_action	Wrapper for timer variable start, pause, clear, and reset commands.
timer_start	Start a timer variable, optionally with a duration payload.
timer_pause	Pause a timer variable.
timer_clear	Clear a timer variable.
timer_reset	Reset a timer variable.
runtime_enable	Enable a runtime-gated item by namespace and id.
runtime_disable	Disable a runtime-gated item by namespace and id.
runtime_set_enabled	Set a runtime-gated item enabled/disabled from payload.
runtime_toggle	Toggle a runtime-gated item.
runtime_enable_all	Enable all runtime-gated items in the requested namespace.
runtime_disable_all	Disable all runtime-gated items in the requested namespace.
enable_action_trigger	Enable one action trigger at runtime.
disable_action_trigger	Disable one action trigger at runtime.
toggle_action_trigger	Toggle one action trigger at runtime.
enable_logic_rule	Enable one logic rule at runtime.
disable_logic_rule	Disable one logic rule at runtime.
toggle_logic_rule	Toggle one logic rule at runtime.
enable_output_sequence	Enable one output sequence at runtime.
disable_output_sequence	Disable one output sequence at runtime.
toggle_output_sequence	Toggle one output sequence at runtime.
enable_input_sequence	Enable one input sequence at runtime.
disable_input_sequence	Disable one input sequence at runtime.
toggle_input_sequence	Toggle one input sequence at runtime.
enable_action_trigger_all	Enable all action triggers at runtime.
disable_action_trigger_all	Disable all action triggers at runtime.
enable_logic_rule_all	Enable all logic rules at runtime.
disable_logic_rule_all	Disable all logic rules at runtime.
enable_output_sequence_all	Enable all output sequences at runtime.
disable_output_sequence_all	Disable all output sequences at runtime.
enable_input_sequence_all	Enable all input sequences at runtime.
disable_input_sequence_all	Disable all input sequences at runtime.
set_state	Set runtime state (e.g., ready, solved).
logic_rule_match	Force a logic rule match/unmatch state.
input_sequence_complete	Emit/complete a sequence event.
sequence_event	Alias for sequence-related event dispatch.
command	Pass through raw command-router JSON.
relay	Control relay output(s).
analog	Control analog output(s).
sound	Control sound playback/effects.
display	Control display output.
light_effect	Apply lighting effect payload.
pixel_overlay	Apply temporary pixel overlay effect.
meter	Meter-style light/level rendering action.
actuator	Control actuator output(s).
sensor_polling	Available action; target and payload depend on your configured modules and integrations.
servo	Control servo output(s).
servo_anim	Available action; target and payload depend on your configured modules and integrations.
output_sequence	Run an output sequence.
action_trigger	Trigger configured action by name/id.
operator_control	Trigger operator control by label/id/index.
collection_reset	Available action; target and payload depend on your configured modules and integrations.
collection_set	Available action; target and payload depend on your configured modules and integrations.
collection_clear_item	Available action; target and payload depend on your configured modules and integrations.
dmx_scene	Recall DMX scene.
hx711_tare	Tare a configured HX711 load cell module.
hx711_calibrate	Persist HX711 calibration data for a configured load cell module.