Myro:Home Helper

Savant Controller Guide

Version 5.50

Local User sign-in, room discovery, live media feedback, controls, and cross-controller automation.

Myro Control, LLC | myrocontrol.com

Contents

Connect your Savant system

- Remembering the Local User

Rooms, activities, and live feedback

- Available Helper services
- Automatic room photos

Remote access with Tailscale

- Required network services

JavaScript API Reference

- MyroSavant
- Controller status and cross-controller scripts

Event-Based Scripts

- ScriptEvents.onSavantRoomChange

AI references, HTTP API, and MCP

Troubleshooting

Connect your Savant system

Helper 5.50 supports Savant as a separate controller alongside HAI / Leviton and NESS / ELK. It connects directly to the Savant host on your local network. This integration is a preview of Savant's app protocol; features depend on your installed host software and configured equipment.

Use a Mac running macOS 12 or later, on the same trusted network as the Savant host or connected through a suitable VPN. Have a **Savant Local User** and that user's local password ready. These are the users advertised by your Savant system, rather than the Mac mini's macOS administrator login. A Savant cloud account password may differ from the Local User password.

1. Click **+** beneath Helper's controller list and choose **Savant (Preview)**.
2. In **Connection**, give the system a name. Choose **Discover...**, or enter the host address and choose **Verify Host...**.
3. Check the host ID, home ID, and certificate fingerprint against the intended system before trusting it. Helper discovers the current WebSocket port.
4. Select a **Local User**. Use **Refresh Users** if the list is empty or the host's users have changed.
5. Enter that user's local password, then choose **Sign In & Connect**. Leave it blank only for a user configured without a required password.
6. Open **Rooms** to see the discovered rooms, sources, and reported state. Choose **Refresh** after installer changes; Helper also refreshes rooms and source commands on reconnect and every five minutes while connected.
7. To let Myro:Home Pro use this Helper for the Savant controller (its Timeline, room photos and finding the Helper on your network), scroll to **Pair with Myro:Home Pro** at the bottom of **Connection**. In Pro, open the controller's settings, choose **Pair with Helper** and scan the QR code or type the code. **Copy** puts the code on the clipboard; **Regenerate...** issues a new code, after which every phone must pair again. The code is for Helper features only: Pro keeps its own direct Savant sign-in and never connects to the host with it.

Local sign-in needs **no cloud token or agent ID**. Account authentication and control use an encrypted WebSocket pinned to the certificate you approved. Room/source inventory is read from the same host's local HTTP configuration service without sending the user's password to it.

Remembering the Local User

Remember Local User in Helper Keychain saves credentials only after the host accepts the sign-in. You can also check it while connected to save the authenticated account immediately. Switching to another controller and back restores the choice; restarting Helper restores it from saved credentials. Checking it before a first sign-in keeps the choice in memory until authentication succeeds.

Unchecking it removes the saved login without disconnecting the current session. The session can reconnect while Helper remains open using its in-memory login, but it will not save that login again. A failed Keychain change is reported and the checkbox returns to its previous state. The displayed password is a fixed mask, never the actual stored password.

Enable **Connect automatically when Helper starts** separately if desired. Automatic startup requires a remembered Local User. **Connect Remembered User** reconnects using the saved account. Removing the controller removes its Helper credentials and cached inventory. Trusting a different host identity clears the old

host's saved credentials and room cache.

Rooms, activities, and live feedback

The **Rooms** pane lists room inventory and available AV services. Select a room, then select a service and use **Start Source**. **Play**, **Pause**, **Stop**, **Next** and **Previous** target the selected active source only. **Room Off** stops all sources confirmed active in the room; volume and mute remain room controls. Source buttons require a successful command discovery and are disabled for commands the source does not advertise. Unavailable command metadata stays unknown; choose **Refresh** to retry. Helper never guesses an active source when several are running.

Two activities can have the same display name while representing different services, such as a console's game, CD, or DVD functions. The source picker and HAP choices show the service type and variant beside the alias. The room picker adds the service ID if those labels still collide. Inspect the service `type` and use its discovered ID in scripts when names repeat. Helper never guesses which duplicate source you meant.

Reported activity, volume, mute, and timestamps arrive over a persistent state subscription. Unknown values stay unknown. Connection loss marks feedback stale, and reconnect establishes a new baseline before later changes trigger scripts. Cached inventory is useful for discovery but does not prove a room is currently on or reachable.

Commands are sent once on an authenticated connection. They are not queued across reconnects or replayed after an interruption. A queued or acknowledged request is not proof the physical device changed; use subsequent room feedback and validate behavior on your installation.

Available Helper services

Feature	Savant behavior in Helper 5.50
Connection and Rooms	Local sign-in, discovery, live feedback, and media controls
Scripting	`MyroSavant`, room-change events, shared Helper integrations, and permitted cross-controller handles
AI reference	Full/compact Savant contract plus actual room/service IDs; mixed homes retain each family's contract
HTTP status	Native Savant connection, room inventory, observed state, and capabilities
Photo library	Available Savant room photos download automatically into this controller's **API > Photos** library
MCP	Shared script-management/reference tools can author Savant automation with the required permissions
HAI/ELK panel controls	Panel objects, security, panel logging, and panel events do not apply to Savant
Browser Rooms	Activities synced from PRO appear in shared HAI/ELK rooms; start/stop only when the matching Savant installation/route is uniquely configured with fresh feedback and discovered commands
HomeKit bridge	Select Savant activities in the **HAP** tab to expose feedback-driven switches in Apple Home

Feature	Savant behavior in Helper 5.50
Myro:Home Pro pairing	**Pair with Myro:Home Pro** on the **Connection** tab shows the code (QR or typed) that lets Pro use Helper's Timeline and room photos for this controller and find the Helper on the network. It is Helper's own code, not a Savant credential
Browser view and Apple TV	Helper lists a Savant controller for browser sign-in with its Control Login or an API key, even when the Helper has no HAI or ELK controller, with Overview, Timeline, Rooms, Activities and Cameras pages; **Run** and **Stop** start and stop one activity. Being listed also lets Myro:Home Pro for Apple TV pair with it, using a six-digit code you allow on the Mac. Security and panel events are not available. These are browser and paired-app surfaces, not scripting APIs; see the Getting Started guide

In browser Rooms, HomeKit assignments and unavailable Savant activities are hidden during normal viewing and remain visible with explanations in edit mode. They count as assigned items and survive edits in Classic. Update PRO and Helper, then use Edit > Sync with Helper in PRO to review and apply source identities. Discovery does not sync shared data automatically. Tapping Edit makes one read-only sync-status check and may put a dot on Sync with Helper; opening a tab does not check or show a banner. Choose Both ways, From Helper or To Helper. Conflicts offer Helper, this device or Skip. Ignore suppresses the dot until Helper changes again; Turn off disables checks until Turn on. There is no background timer or automatic apply. Paired PRO sync introduces source routes; browser editors can reuse existing shared sources but cannot invent or retarget them. Only already-assigned activities can be controlled through the signed-in room; there is no arbitrary destination/command endpoint. See Shared Rooms and Cameras for the browser contract. The Rooms/Cameras sliders icon reveals Add/Edit, photo and Show in Helper options; hidden items remain available in edit mode.

PRO can connect directly to a Savant system with Helper optional. Its room presentation and photo choices are configured in PRO. Adding Savant to Helper does not automatically merge rooms in PRO. HomeKit activities are opt-in through the **HAP** tab.

Savant activities in Apple Home

1. Connect to the trusted Savant host and load its rooms.
2. Open the controller's **HAP** tab, check the activities to expose (up to 149), optionally enter a **Custom Name** for Apple Home and Siri, enable HomeKit, and click **Apply**.
3. In Apple Home, choose **Add Accessory** and scan the displayed QR code or enter its pairing code.

Each selected activity becomes a switch named **Room — Activity**, with the service type and variant for clearer identification. ON starts that activity. OFF stops only that activity; sending OFF to an inactive activity leaves other room activities running. Switch values follow Savant feedback, including changes made with a Savant remote or app. Commands do not fabricate state and are never replayed after reconnecting. Missing, stale or disconnected feedback reports **No Response**.

Custom Name uses the same terminology as HAI and ELK. For example, name an activity **Movie Room** instead of a long installer label. Leave the field blank to use the Savant room/activity label. Apply saves the override in Helper; it does not rename the Savant activity or change its route. The name survives host

uploads, installer renames, reconnects and Helper restarts. Names must fit 64 UTF-8 bytes with no control characters. If you previously assigned a separate name in Apple Home, edit that name in Home too; Home may retain its own override. No pairing reset is needed.

Selection edits preserve the bridge's pairing and existing activity identities. If a host upload regenerates activity UUIDs, Helper uniquely matches the saved component, logical component, variant and activity type within the same room and trusted system, preserving the switch identity. Missing or ambiguous matches remain unavailable. A different trusted host requires a new selection and **Apply**; Helper never silently retargets old switches.

Known older service UUIDs used by scripts and API commands also recover from stored routing metadata, only when there is one match in the same trusted installation and room. Missing or ambiguous routes remain unavailable.

Use **Reset Pairing...** only when you need to remove this bridge's Apple Home pairings. Activity selections stay saved; remove the old bridge from Apple Home and pair again with the new code.

Automatic room photos

After local sign-in, a successful room discovery downloads each available Savant room photo once into **API > Photos** for this Helper controller. Reconnects and app restarts reuse saved photos without downloading those images again. Newly discovered rooms get their first download. Explicitly choosing **Refresh Rooms** downloads existing photos again to check for changes. Helper requests the original size and saves a full-resolution, normalized JPEG plus a thumbnail; it strips source metadata using the same limits as manual photo imports. No Helper API key or cloud token is needed for these downloads.

Each photo is matched by the Savant system and room ID. Reconnecting does not create duplicates; an updated image replaces the same library entry. A rename made in Helper is kept. If you replace or remove a photo locally, that choice takes precedence over future automatic imports. Manual imports remain separate, even with matching names. Missing images are skipped without interrupting media controls, and photos already downloaded are kept if a room becomes unavailable or is removed from Savant.

The Photos pane refreshes as images arrive. In PRO, **From Helper** can explicitly copy a library image onto the device using the existing Helper connection. Later changes in the library do not change copies already saved on devices. PRO's direct Savant connection remains independent of Helper.

Remote access with Tailscale

For setup steps, read Getting Started - Savant Outside the House. PRO's direct Savant connection needs a VPN route to the **Savant host**, independently of any connection to Helper. Tailscale on the Helper Mac alone does not provide that route or proxy PRO's room controls. A Tailscale subnet router can make the host's LAN address reachable without installing anything on the Savant host.

Use the host's known, routed LAN IP, the expected host/home identity and certificate, and the same Savant Local User. Automatic LAN discovery and `.local` names may not work remotely. No cloud token or agent ID is introduced by using a VPN. Helper running at home keeps its existing local Savant connection; it does not need a different host address just because your phone is away.

Required network services

The current Helper and PRO clients use these services on the configured Savant host. If you restrict traffic in your VPN policy or home firewall, allow the actual services used by your installation over the private connection:

Service	Current client behavior
Directed discovery	UDP `9101` to the configured host, to obtain its current control-session port
Inventory and resources	TCP `3060` by default; reads room/source configuration and room photos. A configured HTTP-port override must be reachable too
Authenticated control and live state	TLS WebSocket over TCP at the port returned by discovery; do not assume a fixed control port
Optional Helper API access	Helper Mac TCP `9393`, authenticated with a Helper API key; this is a separate destination from Savant

Reaching only the WebSocket port is insufficient for a fresh discovery/inventory connection. Reaching only Helper is insufficient for PRO's direct media connection. Keep these services off the public Internet and retain certificate verification. Tailscale route approval, access rules, and an online subnet router are all required for the routed-LAN arrangement; an exit node is a different feature.

Validate while the phone is on cellular with Tailscale connected: verify the saved host can reconnect, room/source inventory loads, and fresh state arrives. If you deliberately issue a test command, check its physical result and later reported state.

JavaScript API Reference

- Savant is a preview. Room IDs and service UUIDs come from discovery; service UUIDs may change after a host upload. Helper retains observed routing descriptors (component, logical component, variant and service type), bound to the trusted installation and room, so known old script IDs resolve only when exactly one current source matches. Known older service UUIDs recover without guessing a new source. Unknown IDs, missing routes and ambiguous matches fail; refresh `rooms()` / `services(room)` before generating new scripts. Recovery sends no device commands and does not turn reconnect baselines into automation events. Never resolve repeated source names by choosing the first match.
- HAP configuration is opt-in in Helper's native HAP pane; there is no MyroSavant method or dedicated MCP tool to configure it. Selected switches use reported fresh Savant state and preserve their identity when a unique activity route recovers. The optional Custom Name in that pane sets the Apple Home/Siri label; it does not change a Savant alias or route.

Use the shared Helper Scripting Guide for script lifecycle, shared integrations, schedules, and permissions. The API Reference window marks Savant-specific functions with a **Savant** badge.

MyroSavant

`MyroSavant` belongs to the script's own Savant controller. Room and service arguments accept discovered IDs or unambiguous names, compared without case sensitivity. Names that resolve to more than one item fail. Use the IDs returned by discovery for repeated source names.

Method	Result and behavior
`rooms()`	Cached `{id, name}` room entries
`services(room)`	Cached AV entries with `id`, `name`, `type`, `label`, `variantID`, `commandsAvailable`, and optional `commands` / `commandsUpdatedAt`
`status(room)`	Observed room state, or `null` if unavailable
`start(room, service)`	Request the selected AV activity
`off(room)`	Stop all sources confirmed active by fresh room feedback; a confirmed off room sends nothing
`volume(room, level)`	Request an integer level from 0 through 100
`mute(room, muted)`	Request an absolute Boolean mute state
`play(room)`, `pause(room)`, `stop(room)`	Transport requests for exactly one confirmed active AV source; multiple active sources fail
`next(room)`, `previous(room)`	Skip requests when advertised by exactly one confirmed active source
`transport(room, service, action)`	Target a specific source: `play`, `pause`, `stop`, `next`, `previous`, or `off`. Transport requires that source to be active; `off` leaves other sources alone

- Discovered commands are read-only names, not an arbitrary-command API. Unknown or unsupported source commands fail before transport. All sources in a Room Off batch are validated before any command is sent. A refresh failure disables source commands until inventory is successfully refreshed. Room-only transport methods reject ambiguous source selection; use `transport(room, service, action)` with an exact service ID.
- Command methods return `true` when validated and queued, not when physical completion is proven. Check script logs for send/failure receipts and use observed feedback. Commands are never replayed after reconnect.
- `status(room)` includes `roomID`, `roomName`, `fresh`, `propertyUpdatedAt`, and `inventory` refresh status. Reported optional fields include `activeService`, `activeServices`, `activeAudioService`, `activeVideoService`, `volume`, `muted`, `isActive`, `hasAudio`, `hasVideo`, and `updatedAt` (ISO date). Unknown optional fields are omitted. Inspect `fresh` and each relevant property's timestamp before relying on a reading. Activity strings may be full service identifiers rather than display names.

```
// Read inventory first; use the actual IDs in command scripts.
MyroSystem.log(JSON.stringify(MyroSavant.rooms()));
MyroSystem.log(JSON.stringify(MyroSavant.services("Theater")));
MyroSystem.log(JSON.stringify(MyroSavant.status("Theater")));

// Replace with discovered IDs before enabling a command:
// MyroSavant.start("ROOM_ID", "SERVICE_ID");
// MyroSavant.transport("ROOM_ID", "SERVICE_ID", "play");
```

Controller status and cross-controller scripts

`MyroController.getStatus()` reports `backend: "savant"`, connection, inventory, and capabilities. `refreshStatus()` queues a native inventory refresh; its queue marker is not a correlated device-command acknowledgement. `getRefreshStatus()` reports connection/status, room count, the latest state update, and inventory refresh status.

- `MyroControllers.list()` identifies configured controllers and their backend. `MyroControllers.get(guid)` uses an exact controller GUID and returns `null` without access. For a Savant destination, use the handle's `.savant` bridge with the methods above. A Control4 destination uses the handle's `.control4` bridge instead (see the Control4 Controller Guide); Control4 has rooms and activities, with no panel objects. The origin must grant **Send To** and the destination **Accept From**. Grants are checked on every call, including calls through a retained handle; an MCP client cannot change these local grants.

```
// This script may belong to HAI, ELK, or another Savant controller.
// Configure both local grants and replace the GUID before enabling it.
var destination = MyroControllers.get("SAVANT_CONTROLLER_GUID");
if (destination) {
    MyroSystem.log(JSON.stringify(destination.savant.rooms()));
    // destination.savant.start("ROOM_ID", "SERVICE_ID");
}
```

- Panel `MyroAPI`, `MyroAudio`, and `MyroSensors` functions do not control Savant rooms. Read the destination's own controller reference before invoking its panel APIs. Cross-controller handles issue permitted reads/commands; they do not register room-change events on the destination.

Event-Based Scripts

ScriptEvents.onSavantRoomChange

Method Signature:

```
ScriptEvents.onSavantRoomChange(room, handler)
```

- `room` is a stable room ID, an unambiguous name, or `"*"` on the script's own Savant controller. The callback receives `(roomName, state)`. Registration returns `false` for an unresolved room or a non-Savant controller.
- Initial snapshots and reconnect baselines do not fire change actions. Subsequent normalized changes run on the script queue. Handlers are removed when the script is stopped, disabled, or cleared. Connection events are available through `ScriptEvents.onConnectionChange`; panel zone/area events do not become Savant events.

```
ScriptEvents.clear();
ScriptEvents.onSavantRoomChange("Theater", function (roomName, state) {
    MyroSystem.log(roomName + ": " + JSON.stringify(state));
    // Call a configured Helper integration here, such as MyroVestaboard.
});
```

Shared Helper services, such as Vestaboard, Sonos announcements, MQTT, network requests, and locally approved Mac shortcuts, remain available when configured. Use their exact signatures from the shared guide.

AI references, HTTP API, and MCP

Open the Savant controller's Scripting window, then choose **Help > Copy AI Scripting Reference**, **Copy AI Reference (Compact Text)**, or **Save AI Scripting Reference...**. The reference includes the Savant contract and actual room/service inventory. A missing required controller guide makes reference generation fail rather than substituting another family's instructions.

The keyed HTTP `GET /helper/{controllerID}/status` endpoint returns `controllerFamily: "Savant"`, `backend: "savant"`, native connection state, `rooms`, `roomStates`, `objectCounts`, `inventory`, `homeKit`, and `capabilities`. `capabilities.hap` means support, not that a bridge is configured or running. `homeKit` separately reports `configured`, `enabled`, `bindingMatches`, `selectedActivities`, `running` and `phase` (`stopped`, `waitingForNetwork`, `running`, `failed` or `settingsUnavailable`). No pairing credentials are included. `inventory` reports `available`, `refreshing`, optional `updatedAt` / `error`, `services` and `servicesWithCommands`. Cached command lists do not prove current availability; use the inventory and connection status together. `GET /helper/{controllerID}/scripts/reference` returns the same scoped reference used by Help. Authenticate with a Helper API key for that controller, not the Savant Local User password.

MCP's existing script/reference tools can read this contract and prepare Savant scripts. Saving or enabling scripts requires the Helper API key's **Remote script management** permission. Review new scripts while disabled before enabling them. There are no dedicated Savant MCP media tools or public raw Savant-command endpoint in this release. HAI/ELK `/control`, `/security`, numbered objects, and legacy audio routes are not Savant command contracts.

See the HTTP API guide and MCP guide for authentication and script-management details.

Troubleshooting

Symptom	Check
Host not discovered	Verify the Savant host's current LAN address, then use Verify Host; check network/VPN reachability
Local User missing	Refresh Users on the trusted host; confirm the user's local access in Savant
Invalid password	Use that Savant Local User's local password, not the Mac administrator password; a cloud password may differ
Remember checkbox cannot be saved	Read Helper's Keychain error, then retry while connected; startup auto-connect needs a successfully remembered user
Duplicate source names	Inspect `services(room)` and select the exact service ID and type
Activity unavailable after a host upload	Refresh Rooms or reconnect. Known routes recover automatically; inspect the current source when the match is missing or ambiguous
Transport disabled or refused	Select an active source, inspect its advertised commands, then Refresh; room-only calls require exactly one active source
Missing or stale state	Check connection and timestamps, then reconnect; unknown feedback is not an off/zero value
Cross-controller call refused	Verify the exact GUID, destination backend, and both Send To / Accept From grants
Certificate changed	Verify the intended host identity before trusting a replacement; do not bypass a pin mismatch

For HAI / Leviton or NESS / ELK setup, use their separate controller guides from Help. For general startup and integrations, use Getting Started.

Shared record and photo maintenance

API > Rooms & Cameras appears when the selected controller has shared catalog records. Inspect, rename, hide or remove records there, or open the full web editor. **API > Photos > Combine Exact Duplicates...** previews byte-identical stored images before combining their storage. Existing IDs, photo assignments and independent Savant source metadata remain valid; one source can update later without changing another source's image. This does not change activity discovery, Savant native rooms or HAP selections.