

ohmSTAR Software Guide

The Companion app, command line, HTTP API, Python and AI control for the ohmSTAR and ohmSTAR+

The Decade Box on an Ego Trip

Contents

00 The ohmSTAR Companion app
Install, connect, every page at a glance

01 Choose a route
Which way to drive the box from a computer

02 Command line
The ohmstar tool, and the box's serial commands

03 HTTP API
Endpoints, replies and status codes

04 Python
The ohmstar package

05 AI control
The built-in assistant, and MCP for your own AI app

06 Classroom lab
Several boxes, mock boxes, graded runs

07 Troubleshooting
Messages and fixes

Each chapter is also printed on its own. Using the box itself: see the ohmSTAR Operation Manual. Calibrating with the Companion: see the ohmSTAR Calibration Guide.

00 The ohmSTAR Companion app

AT A GLANCE

For	Running an ohmSTAR or ohmSTAR+ from a desktop: set values, simulate sensors, sweep, run tests, log.
You need	A Windows, macOS or Linux computer, the box's USB cable, the box switched on.
Try it	Plug in the box. The app finds it and connects. On the **Device** page, type `10k` in **Set resistance** and press Enter. The box's LCD shows `10000.00` (ohmSTAR+) or `10000.0` (ohmSTAR). No box? Press **Mock ohmSTAR+** first.

Install

Download: [download link]

System	File	First start
macOS (Apple Silicon)	`.dmg` or `.zip`	Not notarised. Right-click the app, **Open**, confirm.
Windows	`-setup.exe`	Not signed, so SmartScreen warns.
Linux	`.deb` or `.AppImage`	See the port access steps below.

Linux port access

- `.deb`: installs the udev rule `99-ohmstar.rules` for you.
- AppImage: copy the rule yourself: `sudo cp 99-ohmstar.rules /etc/udev/rules.d/ sudo udevadm control --reload-rules && sudo udevadm trigger`
- Not logged in at the computer itself: `sudo usermod -aG dialout "$USER"`, then log out and in (`uucp` on Arch).

Connect

1. Plug the box in and switch it on. The app connects by itself.
2. The top bar shows **Connected**, model, firmware, port and USB chip serial, with **Disconnect**.
3. Not found: press **Detect** in the top bar; it says why. Or, on the Device page, **Search for a box**, or **Choose a port** and **Connect**.
4. No box: **Mock ohmSTAR** or **Mock ohmSTAR+**.

Serial link	FTDI FT232R, 9600 baud, 8N1
Ports searched	FTDI only: USB `0403:6001`, `6014`, `6015`. Nothing is sent to other ports.
Port names	Windows `COMn` · macOS `/dev/cu.usbserial-*` · Linux `/dev/ttyUSB*`

> **One program per box.** The app holds the port while connected. Press **Disconnect** before using the command-line tool, Python or another serial program.

The pages at a glance

Page	What it does
Device	The box's drawn front panel with its live value and keypad. **Set resistance** has two tabs: **Ohms** (`4700`, `10k`, `1.5M`, or a typed list) and **Temperature** (a sensor and a value). Folded below: **Quick values** (memories 0–9, quick keys A–D), **Table upload and download** (slots 1–9, read back and compared), **Box settings** (standby delay, display contrast, restart, firmware). A console shows every command sent, and takes your own.
Compact window	**Compact** in the top bar: only the box and Set resistance or temperature. Tick **Keep on top**. **Full window** returns.
Sensor library	128 curves: RTDs, thermistors, strain gauges, potentiometers. Search, plot, send a curve to the box as a table.
Temperature to ohms	Pick a sensor, enter a temperature (°C or °F), strain or position; the box outputs the matching resistance.

Page	What it does
Custom tables	Build a table from a formula (RTD Callendar-Van Dusen, thermistor Beta or Steinhart-Hart, strain gauge and more), a table file or two-column CSV, or a library sensor. **Run table** steps the box through its points; **Send to box** stores it in a slot.
Sweeps	One page. Mode: **Resistance** (Ω), **Temperature** (library sensor, °C/°F) or **Strain gauge** (με; R0 120, 350, 1000 Ω or custom, gauge factor, lead resistance). Shape: **Steps**, **Hold**, **Step load**, **Ramp**, **Cycles** (triangle or sine), **List**. Up to 5,000 values a pass, 1–100,000 loops, hold 0.3 s to 24 h.
Sequences and reports	Steps of a resistance or sensor value, a label and what to check. Run with **Unit under test** and **Operator**; mark each step Pass, Fail or Skip with a reading and note. Each finished run is saved; **Save PDF** or **Save CSV**.
CSV logging	Every value sent, one file a day: `ohmstar-YYYY-MM-DD.csv`. Columns: `time,box,port,ohms,source,table,table value,table units`. Graph or File view.
Command-line tool	A command line for the connected box. Chapter 02.
Multiple boxes	Several boxes at once, set one by one or all together. Chapter 06.
Local network API	HTTP and MCP server for scripts and AI clients. Chapters 03 and 05.
AI assistant	Ask a language model to work the box; it asks before each change. Chapter 05.
Settings	**Appearance**: Dark, Light or Match this computer. Presets: Everything, Just the box, Standard, HVAC and sensors, Test automation, Cal lab. Each module switches on or off. Always on: connecting, setting a resistance, the console, the mock device.
Calibration bench	Off until switched on in Settings; then a **Calibration Bench** tab appears at the top. See the *ohmSTAR Calibration Guide*.

Where files go

Kind	Folder	File
Sweeps	`Documents/ohmSTAR Companion/Sweeps`	`.ohmsweep`
Sequences	`…/Sequences`	`.ohmseq`
Reports	`…/Reports`	`.ohmreport`
Tables	`…/Tables`	`.ohmtable`
CSV logs	`logs` in the app's user data folder	`.csv`

Plain JSON, one file per item; back them up like any document. Each library starts with an `Examples` folder. Deleting moves a file to the trash.

Rules the app keeps

- Every value is checked against the connected model before it is sent, and rounded to its step. Out of range is refused, never clipped.
- Before anything that changes what the box stores, or a command not in the manual, the app asks first.
- Sweeps, typed lists and sequences won't start while a table is active on the box: values would be read in the table's units.

01 Choose a route

AT A GLANCE

For	Picking how a computer drives the box: by hand, script, test program or AI.
You need	The box on USB. Every route below runs on Windows, macOS and Linux.
Try it	The quickest check that a route works, with no box at all:

```
$ ohmstar set 10k --mock
Set 10 kΩ (10000 Ω) on the mock ohmSTAR.
The box cannot report its output, so check that its LCD shows 10000.0.
This was the built-in mock box, so no real box changed.
```

Routes

Route	Best for	Needs	App open?	Ch.
Companion app	Bench work by hand	The app	Yes	00
In-app command line	Typing commands at a connected box	The app	Yes	02
`ohmstar` tool	Scripts, test stations	Node.js 20+	**No**: disconnect it	02
HTTP API	Any language with HTTP; a second computer	The app, server started, token	Yes	03
Python	Test scripts, tables, sensor values	Python 3.8+, `pyserial`	**No**: disconnect it	04
ohmSTAR Console	Older lab PCs, Windows 7 and up; tables by hand	Nothing to install: unzip and run	**No**: disconnect it	04
AI assistant (in the app)	Plain-language requests	The app, a provider and model	Yes	05
MCP (your AI app)	Claude Code, Codex, Claude Desktop	The app, server, AI control on	Yes	05
Raw serial	LabVIEW, MATLAB, TestStand, a terminal	A serial port, 9600 8N1	**No**: disconnect it	02

Rule of thumb: a person at the screen, use the app or its assistant. A script while the app stays open (logging, other pages), use HTTP or MCP. A script on its own, use the tool or Python. A program that only talks serial, use raw serial and check values yourself.

What each route guards

Guard	App, command line, HTTP, Python, MCP	Raw serial
Value checked against the model, rounded to its step	Yes. Out of range is refused, nothing sent; rounding is reported	No. The box answers `y` to anything
Calibration writes	Not reachable (Python, HTTP, MCP); app and `ohmstar send` ask first	Possible with `T,0,…,D0`
Table writes	Python and the app read back and compare; `ohmstar send` needs `--yes`; HTTP has no route	Possible; check yourself

Rules for every route

- **One program per box.** A serial port opens once: the app, the tool, Python and raw serial take turns.
- **Ranges:** ohmSTAR 1 Ω to 25 MΩ in 0.1 Ω steps; ohmSTAR+ 1 Ω to 5 MΩ in 0.01 Ω steps. Write `4700`, `10k`, `1.5M`; lower-case `m` is refused.
- **The box can't report its output** on firmware 6.1. A program knows only what it last sent; someone at the keypad can change it.

- **Set** `I set` (current protection) before connecting a circuit: at the box, or with the Device page keypad. The tool, HTTP API, Python and MCP have no I set command.

02 Command line

AT A GLANCE

For	Setting values from a terminal or script, and the box's own serial commands.
You need	Node.js 20 or newer and the `ohmstar` tool (`ohmstar.mjs`; not in the app installer: [tool download link]). Or no install at all: the **Command-line tool** page in the Companion.
Try it	No box needed with `--mock-plus`:

```
$ ohmstar set 138.505 --mock-plus
Set 138.51 Ω on the mock ohmSTAR+.
Rounded 138.505 Ω to 138.51 Ω, the nearest step the ohmSTAR+ can make.
The box cannot report its output, so check that its LCD shows 138.51.
This was the built-in mock box, so no real box changed.
```

`ohmstar` below means `node ohmstar.mjs`. The tool and the app can't share a box: press **Disconnect** in the app first. Ports: FTDI only (USB `0403:6001`, `6014`, `6015`); none or several found, the tool lists them and you add `--port`. macOS: `/dev/cu.*`, not `/dev/tty.*`. Linux: the `dialout` group (`uucp` on Arch).

Commands

Command	What it does	Example
`list`	Serial ports that could be a box. Sends nothing.	`ohmstar list`
`info`	Model, firmware, port, USB serial.	`ohmstar info`
`set <value>`	Set the resistance. Checked first; rounding reported.	`ohmstar set 1.5M`
`temp <sensor> <value> [--unit C|F]`	Resistance a sensor has at a value. Strain and position sensors take με and %. Says which sensor it picked.	`ohmstar temp pt100 77 --unit F`
`sensors [search]`	Search the 128-curve library. No box needed.	`ohmstar sensors "10k ntc"`
`table read <slot> [--file f]`	Read user table 1–9; save as a table file.	`ohmstar table read 3 --file t.txt`
`send <line> [--yes]`	One line of the box's serial commands. Anything that changes the box needs `--yes`.	`ohmstar send t,1,2`

Option	Meaning	Exit code	Meaning
`--port PORT`	Use this port. Default: the one FTDI port found.	`0`	Worked
`--mock`, `--mock-plus`	Built-in pretend ohmSTAR or ohmSTAR+.	`1`	Failed or refused
`--json` · `--help`, `-h` · `--version`	JSON result · help, also per command · version	`2`	Typed wrongly

```
$ ohmstar set 10k --json --mock
{"ok":true,"model":"ohmSTAR","modelId":"ohmstar","port":null,"mock":true,"requestedOhms":10000,"ohms":1000
0,"rounded":false}
$ ohmstar set 6M --mock-plus                          # exit 1
Nothing was sent. 6 MΩ is out of reach. The ohmSTAR+ goes from 1 Ω to 5 MΩ.
```

In the Companion, the **Command-line tool** page has a command line for the connected box: `set 10k`, `info`, or a raw line such as `r100` or `t,1,2`. `help` lists commands; Up and Down recall lines. A line that changes what the box stores asks you to confirm. No `--file` there.

Serial commands (firmware 6.1)

9600 baud, 8 data bits, no parity, 1 stop bit. End text commands with `\n` or `\r`. Single-byte commands (`R`, `Z`, `[`, `Q`) take no line ending.

Send	What it does	Reply
`r<ohms>`	Set the resistance, e.g. `r5.123`. With a table active, the value is in the table's units.	`y`
`R`	Open the output. Setting a value closes it again.	`y`
`t,0,<x>`	Read calibration value `x`: 0 = R1 … 30 = R31, 31 = residual.	`t,0,x,value`
`t,0`	Read all 32 calibration values.	One line each
`T,0,<x>,<value>,DO`	**Write a calibration value.** `DO` guards against accidents. Write 0–30, then 31, with no `r` between. Writing 31 reloads settings: the output returns to its last saved value.	`y`, or `G` if refused
`T,<n>,0,<name>` / `T,<n>,1,<units>`	Name table `n` (1–9), up to 20 characters / its units, up to 6.	`y`
`T,<n>,2,-1`	**Erase** its points.	`y`
`T,<n>,3,<value>,<ohms>`	Add a point to the end (max 426).	`y`
`t,<n>,<p>` / `t,<n>`	Read part of a table (`p` 0 name, 1 units, 3+ points) / the whole table.	`t,n,p,value`
`Z` / `[`	Model / firmware version. One byte back: `0x3D` ohmSTAR, `0x33` ohmSTAR+ / `0x61` = 6.1.	One byte
`Q` + one raw byte	One ladder resistor in circuit, 1–31, or 0 for the residual. For calibration. Display unchanged.	`y`
`X7`	Restart. On USB the link drops for about 1 s.	`y`

Rules

- **One command, then wait for the reply.** The buffer is small; overflow is lost silently.
- `y` **means received.** For `r` it doesn't mean accepted: check values first. For `T`, `G` means refused.
- **The reply comes after the last relay pulse** (slowest seen: 300 ms). Let contacts settle before measuring.
- `G`: not understood, or a command arrived in pieces more than 5 s apart.
- **Numbers come back with at most 3 decimals,** fewer for large values, cut off, not rounded.
- **Digit limits.** Largest value with 0 decimals: 268,435,455; 1: 26,843,545.5; 2: 2,684,354.55; 3: 268,435.455; 4: 26,843.5455. Over that, a value silently becomes another number: `r1000000.000` sets about 194,694 Ω; send `r1000000`. Same for table points and calibration values.

Table file example:

```
PRTD_100_385_C
3
0     100
10    103.9
```

```
> Z          < 3     (0x33, an ohmSTAR+)
> r1000      < y
> t,0,31     < t,0,31,0.596
```

> **A bad `r` can put a near-short on your circuit.** `rabc`, empty `r`, `r0` and `r-5` all go to the residual, and the box still answers `y`. Parsing stops at the first bad character: `r10k` sets 10 Ω.

Loading a table

1. At the box, choose no table: `table` `0`.
2. Send `T,n,2,-1`, then the name, units and each point.
3. Read back with `t,n` in the same session. After a power-up `t,n,2` can say 0 although the points are stored.

Table file format

Line 1: name, up to 20 characters. Line 2: decimals shown, 0–5. Then value and ohms in full (no K or M), split by spaces, tabs or commas. Each column only rises or only falls. Comments may follow the numbers. Used by `table read --file`, Python and Custom tables.

03 HTTP API

AT A GLANCE

For	Any program that can send HTTP: set a resistance or a sensor value, read the state, search sensors. The Companion stays open and keeps logging.
You need	The Companion connected to the box, the **Local network API** server started, and its token.
Try it	Replace `YOUR_TOKEN` with the token (**Copy** on the page):

```
curl -X POST http://127.0.0.1:8765/v1/resistance \
  -H "Authorization: Bearer YOUR_TOKEN" \
  -H "Content-Type: application/json" \
  -d '{"value":"10k"}'
{"success":true,"data":{"ohms":10000,"rounded":false},"error":null}
```

Start the server

1. Connect the box on the **Device** page.
2. Open **Local network API** (switch it on in Settings if it isn't listed).
3. Port: `8765`, or any from 1024 to 65535. Press **Start**. The page says **Running**.
4. Token: **Show** or **Copy**. Treat it like a password. **Make a new token** stops the old one at once.

Off until started. The page also has working curl examples with your port.

Connection

Base URL	`http://127.0.0.1:8765`
Listens on	This computer only (`127.0.0.1`). Tick **Allow other computers on the network** for every address.
Auth	`Authorization: Bearer <token>` on every request. Token: 64 hex characters.
Body	JSON, `Content-Type: application/json`, max 64 KB
Rate	100 requests per second per address
Browsers	Any request with an `Origin` header is refused

Endpoints

Request	Body / query	`data` in the reply
`GET /v1/state`		`connected`, `model`, `firmware`, `lastSetOhms`, `activeTable` (`{slot, units}` or `null`)
`POST /v1/resistance`	`{"value":"10k"}`: a string as typed on the Device page, or a number	`ohms`, `rounded`, and `table` if one is active
`POST /v1/sensor`	`{"sensor":"pt100","value":25}`, optional `"unit":"F"`	As above, plus `sensor` (name used) and `value`
`GET /v1/sensors?q=pt100`	Search words	List of `id`, `name`, `maker`, `type`, `min`, `max`, `unit`
`POST /mcp`	MCP, for AI clients	Chapter 05

Every reply: `{"success": true|false, "data": …|null, "error": null|"message"}`. Not offered, by design: raw commands, tables, calibration, opening the output.

```
GET /v1/state
{"success":true,"data":{"connected":true,"model":"ohmSTAR+","firmware":"6.1",
 "lastSetOhms":138.51,"activeTable":null},"error":null}
POST /v1/resistance  {"value":138.505}
{"success":true,"data":{"ohms":138.51,"rounded":true},"error":null}
```

Status codes

Code	`error` text	Cause
200	`null`	Done. `ohms` is what was sent.
400	Send a "value", such as "10k". · Send a "sensor" name and a numeric "value". · That JSON could not be read.	Missing or bad field, bad JSON
401	A valid token is required.	Token missing, wrong or replaced
403	Requests from web pages are not allowed. · That address is not allowed. · AI control is switched off…	`Origin` header; `Host` not local; `/mcp` while AI control is off
404	No such address.	Unknown path
413	That request is too large.	Body over 64 KB
415	Send JSON, with the content type application/json.	Wrong content type
422	The ohmSTAR+ goes from 1 Ω to 5 MΩ. · Use a capital M for megohms, such as 1.5M. · No box is connected.	Value refused; nothing sent
429	Too many requests. Slow down and try again.	Over 100 per second
500	Something went wrong in the app.	App fault

With a table active on the box, the box reads the number in the table's units, and neither POST refuses it. On firmware 7.0 and later the reply then gives `ohms` as what the box says its output is (`null` if it didn't say) and `table` as `{slot, units, asked, value, confirmed}`. Firmware 6.1 can't say a table is active, so `activeTable: null` there proves nothing. A script that means ohms: check `activeTable` first, and on 6.1 make sure the box shows no table.

Allowing other computers drops the local-address check. Anyone on the network with the token can set the box. MCP still answers this computer only.

04 Python

AT A GLANCE

For	Test scripts: set resistances and sensor values, read and load user tables. Talks to the box over USB directly.
You need	Python 3.8 or newer (3.8 is the last for Windows 7), `pyserial` 3.5+, and the `ohmstar` package (0.1.0; not on PyPI yet: [package download link]). The Companion disconnected from the box.
Try it	No box needed: `MockBox` follows the same rules as a real one.

```
from ohmstar import Box, MockBox

with Box(MockBox("ohmSTAR+")) as box:
    print(box.model, box.firmware)        # ohmSTAR+ 6.1
    box.set("10k")
    print(box.last_sent)                  # 10000.0
    r = box.set_sensor("pt100", 25)
    print(r.ohms, r.sensor.name, r.others)  # 109.74 Pt100 a0.00385 2
```

Install and connect

1. `python -m venv .venv`
2. `.venv/bin/pip install <package folder>`
   Windows: `.venv\Scripts\pip`
3. Plug in the box, switch it on, disconnect the Companion.
4. Change one line of the example:

```
with Box.connect() as box:  # the one box
    box.set("4.7k")
box = Box.connect("COM4")   # a named port
```

Port names: `COM4`, `/dev/ttyUSB0`, `/dev/cu.usbserial-A1B2`.

Port	9600 baud, 8N1. FTDI only (`0403:6001`, `6014`, `6015`).
On opening	Only `Z` and `[` are sent. Not an ohmSTAR or ohmSTAR+: refused and closed.
Timeouts	`reply_timeout=1.5` s, `idle_time=0.15` s (keyword arguments)
Threads	One command at a time; threads sharing a `Box` take turns.

Reference

Call	Does
`Box.connect(port=None)`	Open and identify a box.
`Box(MockBox("ohmSTAR+"))`	A pretend box. `MockBox()` is an ohmSTAR+; `MockBox("ohmSTAR")` the other.
`find_ports()`	Lists FTDI ports. Sends nothing.
`box.model`, `box.firmware`	`"ohmSTAR+"`, `"6.1"`
`box.set(value)`	Ohms, or `"10k"`, `"1.5M"`. Rounded to the step, halves up. Returns ohms sent.
`box.last_sent`	Last value this script sent. `None` after a timeout or odd reply. Not a reading.
`box.set_sensor(s, v, unit=None)`	°C, or `unit="F"`; με for strain, % for position.
`ohmstar.sensors("10k ntc")`	Search the 128-curve library.
`Table.of(name, units, points)`	A table; `.problems()` lists faults.
`Table.from_file(p)`, `t.to_file(p)`	The manual's table file format.
`box.upload_table(slot, t, overwrite=False)`	Slots 1–9. Erase, write, read back, compare.
`box.read_table(slot)`	Read a user table.
`box.send_raw(line, allow_risky=False)`	Reads only: `t,…`, `Z`, `[`. Returns bytes.

Exceptions

Problem	Exception
Value out of range, bad slot, unknown sensor	`ValueError`
Bad table	`InvalidTableError`
Raw command refused	`UnsafeCommandError`
No box / several boxes	`NoBoxFoundError` / `SeveralBoxesError`
Port won't open	`PortOpenError`
Not an ohmSTAR	`NotABoxError`
No answer in time	`BoxTimeoutError`
Cable pulled, port closed	`BoxConnectionError`, `BoxClosedError`
Box answered `G`	`BoxRefusedError`
Slot in use	`SlotInUseError`
Upload cut short / differs	`UploadIncompleteError` / `TableMismatchError`

All but the `ValueError` family inherit from `OhmstarError`.

Real replies

```
box.set("6M")
ValueError: Nothing was sent. 6 megohms is out of
reach. The ohmSTAR+ goes from 1 ohm to 5 megohms.
box.set(138.505); box.last_sent  # 138.51
box.upload_table(2, other)
SlotInUseError: Slot 2 holds "Pt100" with 2
points.
Pass overwrite=True to replace it.
box.send_raw("t,0,31")  # b't,0,31,0.6\r\n' (mock)
```

Safety facts

- The box answers `y` to any `r`, even `rabc`. `set` checks first. Lower-case `m` is refused; no scientific notation is sent.
- No calibration method and no general write. `send_raw` refuses calibration writes even with `allow_risky=True`.
- A table name can't be blanked on a real box, so any named slot counts as in use.
- A timeout may mean the box switched itself off. Switch it on and retry.

ohmSTAR Console: the same package, no programming

For an older lab PC, or one where nothing may be installed. Unzip `ohmSTAR_Console_portable.zip` anywhere, even a USB stick, and double-click `ohmSTAR Console.bat` (Windows 7 SP1 and up; it carries its own Python). On a Mac or Linux PC, `ohmstar-console.command` uses the Python 3.8+ already there. Keys: the number or letter shown; B or Esc goes back.

- Set a resistance, or a value on any library sensor; step through a list or a range, timed or by key.
- All nine table slots at a glance, each as a chart; save one to a file, load a file (old table files work as they are), or make one from a sensor at the range and step you choose. Every table sent is read back and checked.
- `--demo` runs it on a pretend ohmSTAR+. Slot 0, the calibration, is never read or written.

Testing your scripts: `mock = MockBox(); box = Box(mock, reply_timeout=0.2); mock.switch_off()` makes the next call raise `BoxTimeoutError`. `mock.state` shows what the mock holds.

05 AI control

AT A GLANCE

For	Working the box in plain words: "Set up a Pt100 and sweep it from 0 to 100 °C in 5 degree steps, 30 s each."
You need	**Built in:** the Companion, and a provider with a model (an online service with your API key, or a model on this computer). **Your own AI app** (Claude Code, Codex, Claude Desktop): the Local network API server running, **Allow AI control (MCP)** on, and the token.
Try it	Claude Code, after the token step below (port 8765). Then type `/mcp` in Claude Code: `ohmstar` is listed.

```
claude mcp add --transport http --scope user ohmstar http://127.0.0.1:8765/mcp --header
'Authorization: Bearer ${OHMSTAR_TOKEN}'
```

Built-in AI assistant

1. Open **AI assistant**. Choose **Provider** and **Model**; paste an **API key** if needed. **Test connection**.
2. Ask. Before any change to the box it shows exactly what it will send: **Allow**, **Allow for this conversation** or **Don't allow**.
3. **Keep beside pages** docks it next to any page.

Online	Anthropic (Claude), OpenAI, Google Gemini, DeepSeek, OpenRouter, Groq, Together AI, Mistral, xAI (Grok)
This computer	Ollama, LM Studio, llama.cpp server, vLLM. Nothing leaves the computer.
Custom	Any OpenAI- or Anthropic-compatible address
Sent to a provider	What you type, tool descriptions, the app's guide, tool replies. Never serial number, dates, USB chip serial or port.
API keys	Sealed by the operating system; each sent only to its own address.
Never asked	Stop sweep, define a custom sensor, make a table, save a sequence
Asked every time	Writing a table into a box slot (only the built-in assistant can)

Your own AI app (MCP)

1. Companion: connect the box, open **Local network API**, **Start**.
2. Switch on **Allow AI control (MCP)**. It is off at every app start unless **Keep on after restart** is ticked.
3. Press **Copy** on the token, then put it in `OHMSTAR_TOKEN`:

```
# macOS, Terminal
echo "export OHMSTAR_TOKEN=$(pbpaste)" >> ~/.zshrc
&& source ~/.zshrc
# Windows, PowerShell (then open a new terminal)
[Environment]::SetEnvironmentVariable('OHMSTAR_TOK
EN', (Get-Clipboard), 'User')
```

Client	Setup
Claude Code	The command above. Remove: `claude mcp remove ohmstar --scope user`
Codex	In `~/.codex/config.toml`: `[mcp_servers.ohmstar]` `url = "http://127.0.0.1:8765/mcp"` `bearer_token_env_var = "OHMSTAR_TOKEN"` Check: `codex mcp list`
Claude Desktop	**Save Claude Desktop extension**, open the `.mcpb` file, enter Token and Port when asked.

Apps opened from the Dock don't read `~/.zshrc`: start Claude Code or Codex from Terminal. Linux: add `export OHMSTAR_TOKEN=…` to your shell profile. Repeat after a new token.

Tools

Tool	Does	Tool	Does
`get_box_state`	Model, firmware, range, output, table, who has the box	`start_sweep`	List or range of ohms or sensor values, each held

Tool	Does	Tool	Does
`search_sensors`	Search the library and custom sensors	`stop_sweep`	Stops at once. Always allowed
`sensor_to_ohms`	Resistance at a value; sends nothing	`get_sweep_status`	Progress, time left, last result
`set_resistance`	Connects the output at `"100"`, `"4.7k"`, `"1.5M"`	`define_custom_sensor`	RTD by R0, thermistor by R25 and beta, Steinhart-Hart, points. Until the app closes
`set_sensor_value`	E.g. a Pt100 at 25 °C	`make_table`, `list_tables`	Save a table file in Tables; list them
`open_output`	Disconnects the terminals (broken wire)	`save_sequence`, `list_sequences`	Save a sequence file; list them
`close_output`	Reconnects, at the old or a new value	`run_sequence`	Starts a saved sequence on the Sequences page
`preview_sweep`	Values and duration; sends nothing	`read_reports`	Newest run reports, step by step

Limits

Sweep	1,000 values a pass, 100 loops, 10,000 values in all
Hold	1 s to 3,600 s a value; 24 h in all
Saved sequence	200 steps
Custom sensors	20, up to 1,000 points each
Values	Same checks as the Device page; refused, never clipped
MCP from	This computer only, even with other computers allowed

Refused or locked away

- **Refused while** the calibration bench, a table upload or a firmware update has the box, a sweep or sequence runs, or a table is active on the box. MCP also while AI control is off.
- **Never reachable:** calibration and its record, serial number and dates, firmware updates, raw commands, factory reset, power-up settings.
- Turning off AI control, stopping the server or closing the app stops what an AI started. **Recent AI actions** on the Local network API page lists what it did.

06 Classroom lab

AT A GLANCE

For	Teaching with several boxes, or with none: sensor curves, controller checks, graded exercises, scripting.
You need	The Companion on each bench computer, or one computer with several boxes on USB. No boxes yet: the mock boxes.
Try it	Two benches simulating a Pt100 at body temperature, in Python, with mock boxes:

```
from ohmstar import Box, MockBox

benches = [Box(MockBox("ohmSTAR")), Box(MockBox("ohmSTAR+"))]
for n, box in enumerate(benches, 1):
    sent = box.set_sensor("pt100", 37)
    print(f"Bench {n}: {box.model} set to {sent.ohms} ohms")
    box.close()
Bench 1: ohmSTAR set to 114.4 ohms
Bench 2: ohmSTAR+ set to 114.38 ohms
```

Lab layouts

Layout	Use
One box and computer per bench	The Companion. In Settings pick a preset: **Just the box** (connect, set, console) or **Standard** (adds Sensor library, Temperature to ohms, Custom tables).
Several boxes on one computer	**Multiple boxes** page: add each by serial port; **Set** each or **Set all**. Or a Python loop, one `Box.connect("COM4")` per port.
Students without a box	**Mock ohmSTAR** / **Mock ohmSTAR+** in the app, `--mock` in the tool, `MockBox` in Python.
Instructor computer driving bench computers	HTTP API on each bench with **Allow other computers on the network** ticked; one token per bench (chapter 03).

Multiple boxes page

Add	**Serial port** and add, or a mock ohmSTAR / ohmSTAR+
Each box shows	Model, firmware, USB serial, last value sent; its own value and **Set**; **Remove**
Set all	One value to every box. A box with a table active is marked: it read the number in table units.
Told apart by	Port, because some units share a USB serial number
Device page box	Has its own connection; not in this list

Values are checked per model: `100.05` becomes 100.1 Ω on an ohmSTAR and stays 100.05 Ω on an ohmSTAR+.

Exercises and where they live

Topic	App feature
RTD and thermistor curves	Sensor library plots; Temperature to ohms
Controller or thermostat check	Sequences: a step per set point, Pass/Fail with the reading
Strain gauge	Sweeps, Strain gauge mode: R0, gauge factor, leads, με
Thermal cycling, ramps	Sweeps: Cycles (triangle, sine), Ramp
Sensor tables on the box	Custom tables, Send to box; then `table` at the box
Instrument scripting	Chapters 02 to 04, with mock boxes

Graded runs

1. Instructor: **Sequences and reports**, new sequence, one step per check (resistance or sensor value, label, what to check). Copy the `.ohmseq` file to each bench's `Documents/ohmSTAR Companion/Sequences`.

2. Student: run it with **Unit under test** and **Operator** filled in; mark each step Pass, Fail or Skip, typing the reading and a note.
3. Hand in **Save PDF** or **Save CSV**. Every run is also kept in `Reports`, and every value sent is in the day's CSV log.

Safe defaults for a class

- **Set `I set` on each box** to the student circuit's voltage before class. 1 W rating.
- **Calibration bench is off** until switched on in Settings. Python, the HTTP API and AI can't reach calibration.
- **At the box,** menu 3 starts calibration and Factory Reset replaces it. Brief students.
- **To stop table writes from the app,** switch off Table upload and download and the Command-line tool. The console still takes raw commands, after a warning.

AI in class

- Use a model on the computer (Ollama, LM Studio, llama.cpp server, vLLM): nothing leaves it.
- The assistant asks before every change to the box and can't reach calibration, firmware or raw commands.
- Sweeps, sequences, the HTTP API and AI work on the box connected on the Device page.

07 Troubleshooting

AT A GLANCE

For	When a computer can't reach the box, or the box does something unexpected.
You need	The message shown. Most messages say what to do.
Try it	First test: identify the box with the Companion disconnected. Here with the mock; on a real box, drop `--mock-plus` and Port and USB serial fill in.

```
$ ohmstar info --mock-plus
Box:        ohmSTAR+, firmware 6.1
Port:       none (built-in mock box)
USB serial: not known
Range:      1 Ω to 5 MΩ, in steps of 0.01 Ω
```

Connecting

Symptom or message	Fix
App: **Box Not found**	Box on? USB cable in? Press **Detect**: it says why. Try another cable or port.
No box found. Check that it is switched on…	As above. `ohmstar list` shows the ports; pick one with `--port`.
More than one box could be connected	Add `--port` (tool) or name the port in `Box.connect("COM4")`.
Could not open … another program may be using it	One program per box. Press **Disconnect** in the Companion, or close it; close terminals and other serial programs.
Not allowed to open … (Linux)	Join `dialout` (`uucp` on Arch), log out and in. AppImage: install the udev rule (chapter 00).
macOS: "unidentified developer"	Right-click the app, **Open**, confirm. Once.
Windows: SmartScreen warning	The app isn't signed yet. Allow it to run.
The calibration bench is using the serial ports	Go back to the **Companion** tab first.
A timeout; the box was idle	On battery it turns itself off after about 5 minutes, and no command wakes it. Switch it on. It stays on while USB is connected to a computer.

Values and replies

Symptom or message	Fix
Nothing was sent. … is out of reach.	Outside the model: ohmSTAR 1 Ω–25 MΩ, ohmSTAR+ 1 Ω–5 MΩ.
Use a capital M for megohms	`1.5M`, not `1.5m`.
Rounded … to …	Normal: the nearest step, 0.1 Ω (ohmSTAR) or 0.01 Ω (ohmSTAR+).
"pt100" matches several sensors / nothing was chosen	Use a fuller name or the id from `ohmstar sensors`, e.g. `Omega/Pt100_a0.00385`.
… is outside the … table, which runs from …	Sensor values stop at the curve's ends; nothing is extrapolated.
The box reads a different number than sent	A table is active: the number was read in table units. At the box, `table` `0`. Sweeps and sequences refuse to start until then.
Raw serial: no reply	9600 8N1? Line ending on text commands? Another program on the port? Try `Z` first.
Raw serial: `G`	Not understood, or arrived in pieces more than 5 s apart. For `T`: refused.
Raw serial: the value is wrong	Too many digits (chapter 02, digit limits), or a `k/M` suffix: `r10k` sets 10 Ω.

Symptom or message	Fix
Exit code 2 from `ohmstar`	Typed wrongly. `ohmstar <command> --help`.

HTTP API and AI

Symptom or message	Fix
Can't connect at all	Companion open? Server **Running** on the port you use? Changed the port: copy the setup again.
401 A valid token is required.	Token wrong, old or not in `OHMSTAR_TOKEN` where the client runs. A new token stops the old one at once.
403 AI control is switched off	Switch on **Allow AI control (MCP)**. It is off at each start unless **Keep on after restart** is ticked.
403 Requests from web pages are not allowed.	Browsers can't call the API. Use a script or tool.
No box is connected.	Connect it on the Device page.
The box is busy	The reply names what has it: bench, upload, firmware update, sweep or sequence. Wait, or stop the sweep.
Claude Desktop doesn't list the tools	Extension on in **Settings › Extensions**? Token and port match? Logs: `~/Library/Logs/Claude` (macOS), `%APPDATA%\Claude\logs` (Windows).
Claude Code or Codex has no token	Apps started from the Dock don't read `~/.zshrc`. Start them from Terminal.

Still stuck: support@labmetric.com. Include the message, the model and firmware (`ohmstar info`, or the Companion's top bar), and the operating system.