

Motion Script Automation

(For ESP32-S3 Robot Driver Boards)

This guide introduces how to use the **Automation Scripts** component to create action sequences without writing code.

By the end, you'll be able to make your servos (or other supported peripherals) move automatically — triggered by buttons or even on boot.

1. Before You Start

If you haven't gone through the [Quick Start Guide](#) yet, please read it first.

It explains the **Servo Control** and **Automation Scripts** modules, which this guide builds upon.

2. Power and Connections

The Automation Scripts modules can control **any function** that the driver board supports via JSON commands — not just servo motion.

If you're controlling servos, make sure to connect **external power** to the driver board.

Power-On Instructions

1. Connect the **User Interface board** to the **Driver board**.
2. Power via **DC / XT30 (2+2)** connector: toggle the switch to **ON**.
3. If a **Type-C** cable is connected, the ESP32-S3 will power on regardless of the switch,
but the DC connector will not power servos if the switch is **OFF**.
4. Power-on indicators:
 - The buzzer beeps once.

- The OLED shows AP/STA IP info. (STA IP will be blank if Wi-Fi is not configured yet.)
-

3. Accessing the Control Page

There are two Wi-Fi connection modes:

Option A: AP Mode (Access Point)

- SSID: **Robot**
- Password: **12345678**
- Pros: simplest setup
- Cons: your device will lose internet connection while connected to the board

Access via browser at `192.168.4.1`.

Option B: STA Mode (Connect to Router)

- Configure Wi-Fi via **Wi-Fi Setting** on the control page.
- Once connected, the `STA Status` will display the assigned IP (e.g., `192.168.0.103`).
- You can access the control page using that IP address from any device on the same LAN.
- Pros: your device stays online and can still control the board.

Recommended browsers: Chrome, Edge, Safari, Firefox.

4. Overview of the **Automation Scripts** Component

AUTOMATION SCRIPTS

Mission upload status.

Input json cmd here.

Up

Down

Left

Right

On Boot

STOP MISSION

Delete Mission

Up

Down

Left

Right

On Boot

This module allows you to visually program sequences of servo movements or other control actions — no coding needed.

AUTOMATION SCRIPTS

Mission upload status.

```

{"T":11,"id":1,"pos":200,"spd":0,"acc":0}
{"T":11,"id":2,"pos":300,"spd":0,"acc":0}
{"T":11,"id":3,"pos":400,"spd":0,"acc":0}
{"T":11,"id":4,"pos":500,"spd":0,"acc":0}
{"T":51,"delay":1000}
{"T":11,"id":1,"pos":1200,"spd":0,"acc":0}
{"T":11,"id":2,"pos":1300,"spd":0,"acc":0}
{"T":11,"id":3,"pos":1400,"spd":0,"acc":0}
{"T":11,"id":4,"pos":1500,"spd":0,"acc":0}
{"T":51,"delay":1000}
{"T":11,"id":1,"pos":200,"spd":0,"acc":0}
{"T":11,"id":2,"pos":300,"spd":0,"acc":0}
{"T":11,"id":3,"pos":400,"spd":0,"acc":0}
{"T":11,"id":4,"pos":500,"spd":0,"acc":0}

```

Up

Down

Left

Right

Each button on the top (**Up** , **Down** , **Left** , **Right**) uploads a different **mission file** to the board and binds it to the corresponding direction on the five-way switch.

Button	Mission File	Description
Up	<code>up.mission</code>	Triggered when joystick is pushed up
Down	<code>down.mission</code>	Triggered when joystick is pushed down
Left	<code>left.mission</code>	Triggered when joystick is pushed left
Right	<code>right.mission</code>	Triggered when joystick is pushed right
On Boot	<code>boot_user.mission</code>	Runs automatically at startup (looped)
STOP MISSION	—	Stops the currently running mission
Delete Mission	—	Removes mission files

You can combine **servo control JSON commands** and **delay commands** to choreograph complex multi-servo movements.

Remember: each line in the script must contain **exactly one JSON command**, or the Web App will fail to parse it.

5. Full Example: Programming Multi-Servo Motion

What You'll Need

- **Robot Driver with ESP32-S3 (A)** ×1
- **12V power supply or 3S Li-ion battery pack** ×1
- **SC-1500 bus servos** ×3
- **Computer / Tablet / Smartphone** (with Chrome, Edge, Safari, or Firefox)
- **Wi-Fi network** (optional but recommended)

Goal

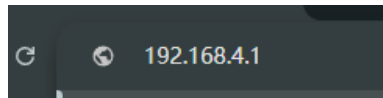
Create a sequence that moves three new SC-1500 servos simultaneously when the joystick is pressed **Up**.

During this process, you'll also learn how to configure Wi-Fi, assign servo IDs, and compose commands.

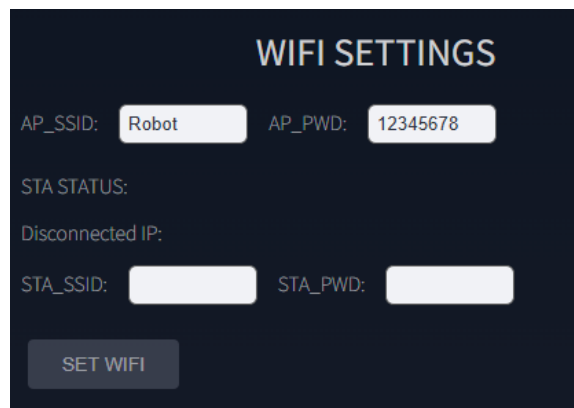
Step 1: Initial Wi-Fi Setup

This step only needs to be done once.

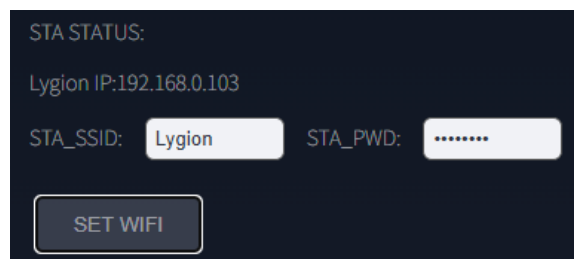
1. Power the driver board with 12V (or 3S battery).
2. Switch ON the power.
3. Wait for the beep and OLED display.
4. Connect to Wi-Fi **"Robot"**, password **12345678**.
Ignore "no internet" warnings — stay connected.
5. Open your browser → go to **192.168.4.1** → enter the control interface.



6. In **Wi-Fi Setting**, enter your router's SSID and password → click **SET WIFI** → confirm.

A screenshot of a "WIFI SETTINGS" screen with a dark background. At the top, it says "WIFI SETTINGS" in white. Below that, there are two input fields: "AP_SSID:" with the value "Robot" and "AP_PWD:" with the value "12345678". Underneath, it says "STA STATUS:" followed by "Disconnected IP:". Below that are two more input fields: "STA_SSID:" and "STA_PWD:". At the bottom, there is a button labeled "SET WIFI".

7. Once connected, note the STA IP shown (e.g., **192.168.0.103**).

A screenshot of the "WIFI SETTINGS" screen after a connection. The "STA STATUS:" section now shows "Lygion IP:192.168.0.103". The "STA_SSID:" input field contains the text "Lygion", and the "STA_PWD:" input field contains a series of dots. The "SET WIFI" button is still at the bottom.

8. Reconnect your computer/phone to your normal Wi-Fi network (the same one you configured above).
9. Visit the IP address you noted — the control page is now accessible without losing internet.

Step 2: Assign Servo IDs

⚠ **Important:** When changing IDs, only connect **one** servo at a time.

New servos all have default ID **1**. If you connect multiple servos at once, they will all be reassigned to the same ID.

1. Connect one servo to the driver board.
2. Select **500K baud rate** in the Web UI (required for SC-1500).




3. Use **Change ID** → enter **4** → click **Set** → confirm.



4. Repeat for other servos, assigning IDs **5** and **6**.

Step 3: Test Servo Movement

Enter a command such as position **200**, then click **Action**.



The screenshot shows a web interface titled "SC SERVO CTRL". It contains several input fields and buttons. The "ID" field is set to 4, and the "Pos" field is set to 200. The "Time" field is set to 0, and the "Spd" field is set to 0. There are buttons for "Release", "Torque", and "Feedback". Below these are "Add" and "Action" buttons, with "Action" being red. At the bottom, there are fields for "Change ID:" (set to 4) with a "Set" button, and "Delay(ms):" with an "Add" button.

You should see the servo rotate.

If not, check:

- 12V power connection
- Switch set to **ON**
- Web uptime counter is running (if not, refresh or check Wi-Fi)
- Correct servo control module selected (e.g., "SCS Servo Control")
- Correct ID (try broadcast ID **254** if unsure)

💡 Note on Servo Range:

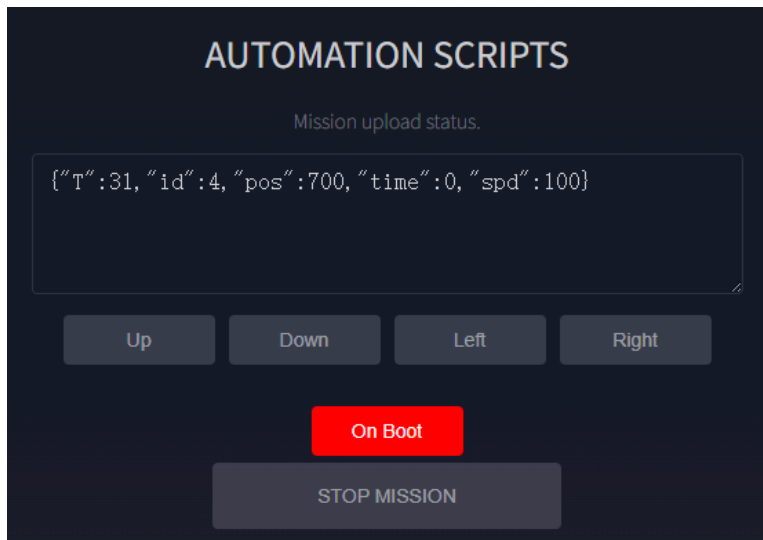
SC-series servos have 0–1023 encoder range but only move within 20–1001 for stability.

That's roughly 210° rotation.

Step-to-angle formula: $\text{steps} = (1024 / 220) \times \text{angle}$.

Step 4: Add Servo Commands

Click **Add** to append the current command to the **Automation Script** input box.
Each line represents one JSON command.



Example command (automatically generated):

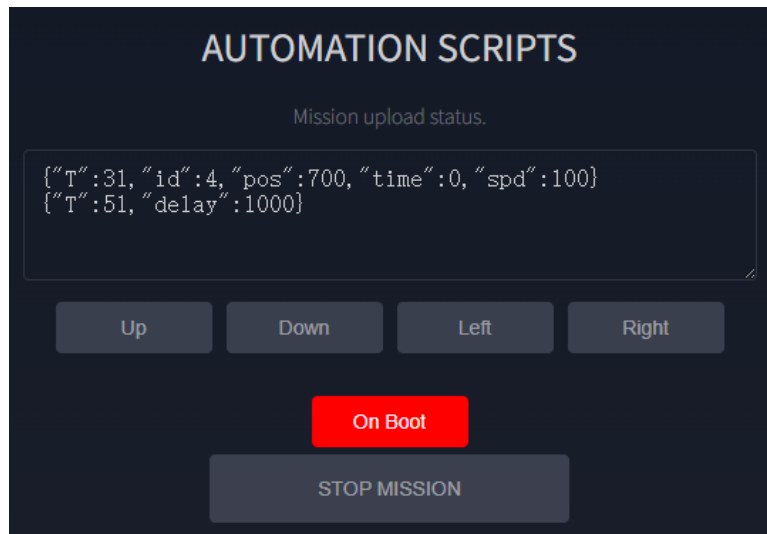
```
{"T":31, "id":4, "pos":700, "time":1000, "spd":0}
```

Add a delay instruction between movements:



```
{"T":1, "time":1000}
```

This adds a 1-second delay between steps. Without it, multiple actions execute instantly, appearing as a single motion.

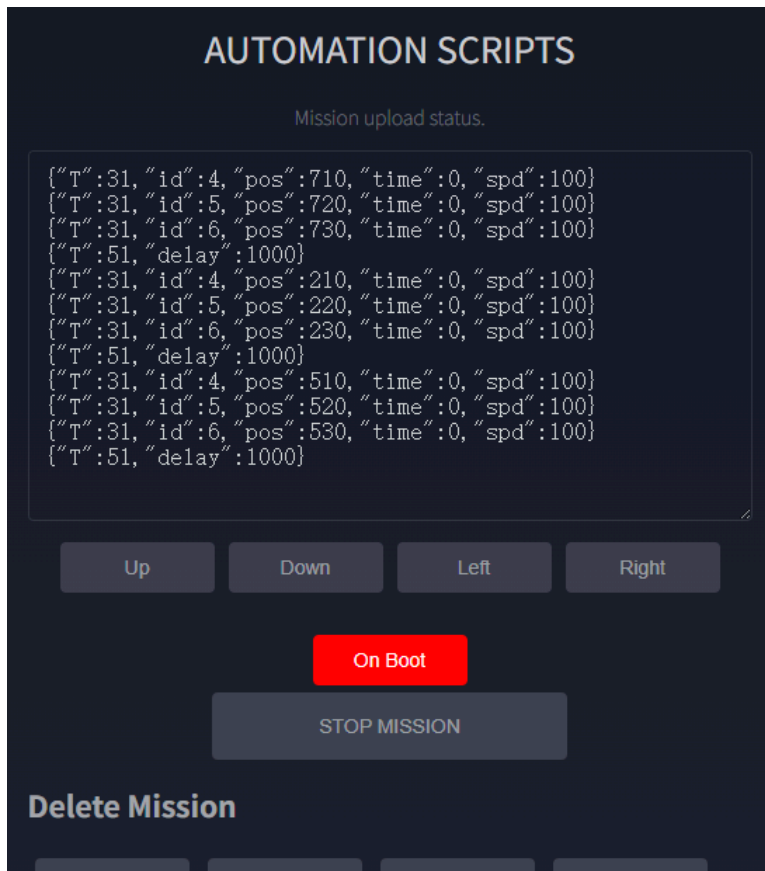


Step 5: Build the Full Script

Repeat the process for servo IDs 5 and 6.

You can copy-paste and adjust parameters manually — enlarge the text box if needed.

Example multi-servo script:



Step 6: Upload and Trigger the Script

- Click **Up** to save the script as `up.mission`.
- Once uploaded, push the joystick **Up** — the sequence runs automatically.
- Mission files are stored in flash memory and persist after reboot.

You can assign different scripts to each joystick direction or even make one run on boot.

Step 7: Advanced Control via JSON

For advanced users, missions can also be created or run directly using JSON commands:

Action	JSON Example
Delete mission	<code>{"T":309,"name":"up"}</code>

Action	JSON Example
Create new mission	<code>{"T":301,"name":"up","intro":"created via web"}</code>
Add command	<code>{"T":303,"name":"up","json":{"...}}"</code>
Run mission	<code>{"T":308,"name":"up","interval":0,"loop":1}</code>
Stop mission	<code>{"T":0}</code>

- `interval` : delay between commands (ms)
- `loop` : number of loops (`-1` for infinite)

`On Boot` creates a `boot_user.mission` that loops automatically after power-on.

You can stop it anytime with **STOP MISSION** or `{"T":0}`.

There's also a system `boot.mission`, used for automatic configurations (e.g., setting baud rate or reconnecting Wi-Fi):

```
{"T":303,"name":"boot","json":{"T":10,"baud":500000}}
```

Step 8: Resetting the System

Click **Reset** to format the ESP32-S3 file system.

This will delete all mission files and stored Wi-Fi configurations — useful for restoring factory state.

6. Summary

By now, you've learned how to:

- Connect and power the driver board
- Configure Wi-Fi in AP or STA mode
- Assign servo IDs and test control
- Create and upload motion scripts visually
- Understand how mission files and JSON commands work behind the scenes

With **Automation Scripts**, you can choreograph complex servo behaviors, trigger them via buttons, or even have them run automatically at startup — all without

writing a single line of code.

Would you like me to format this into a **ready-to-publish Markdown/HTML user manual** (with consistent heading styles, embedded notes, and icons)?

That version would be ideal for your product page or documentation site (like GitHub Pages or your lygion.ai site).