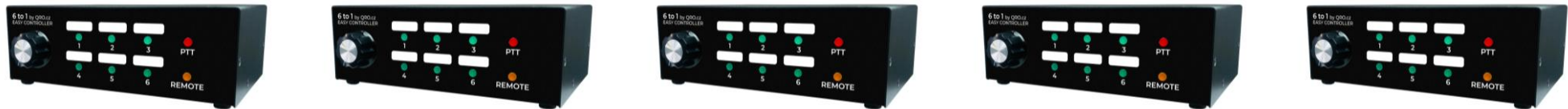
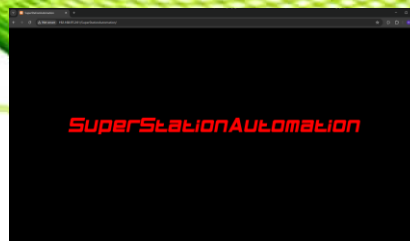


# Super.Station.Automation

LAN / ETHERNET / WAN



LAN / ETHERNET / WAN



## The problem:

YOU need to click on many buttons to set up your desired station: bands, antennas, filters...?

YOU need to click on many buttons to change a current state of your station?

YOU just want an easy way to controll, setup and orchestrate you station by just on click?

①

YOU want to fire different commands with a decent delay and chain them?

YOU want to create customizable dashboards with simple call to action buttons?

②

YOU want to store the individual states of all of your controllers after configuration?

YOU want to recall a saved state of all of your controllers any time:

wpx2021, cqww2023ssb, cqww2025cw, .... Setup done in seconds!

→ All this included and for free for your [qro.cz](http://qro.cz) / [hamparts.shop](http://hamparts.shop) controller! ←

The solution:

***Super.Station.Automation***



## The goal:

- Reduce setup time!
- Reduce time for changing!
- Reduce and hide complexity!
- Increase reliability!
- Save a current complete controllers/station setup in a file or as many as u like!
- Recall stored complete controllerstates/station setups – as many as u like!
- Build as many and as simple dashboards for call to actions/buttons as u like!
- Chain commands of local controllers, of different controllers in ur networks, or even controllers in australia together behind a button!
- On-button-click => Fast, reliable execution!
- The choice is yours: Local App on your computer or put it on your webserver
- API to trigger the buttons by a url
- Write your own Plugins for DXLog, Streamdeck, NodeRed, ...



## What is your setup?

### Lessons learned:

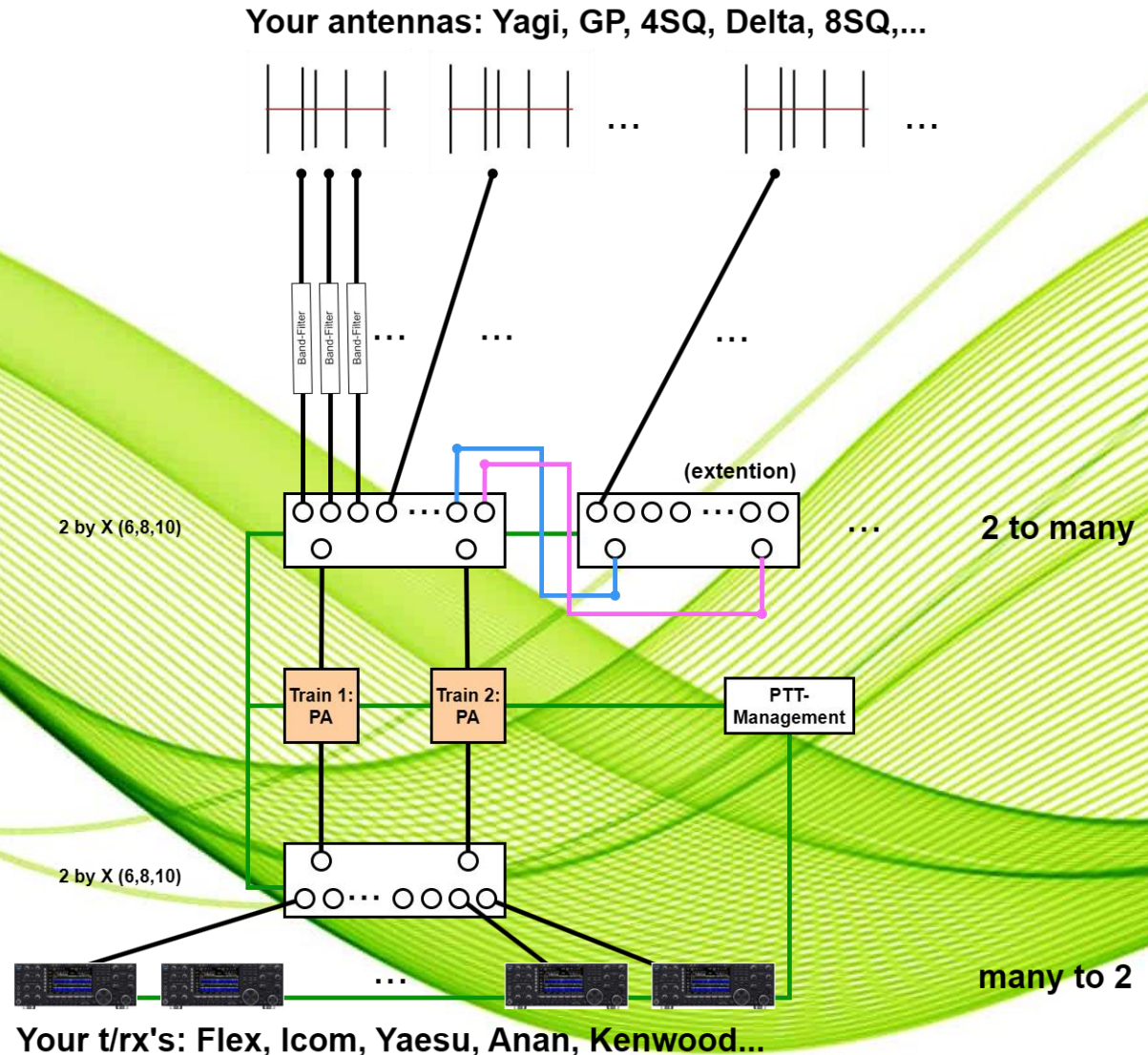
- many have more antennas
- many have even more ants per band
- many have more then 1 trx :p
- many want to have 2 signals on the air
- many have more than one amp
- doing contest in a team is fun

So the most basic setup to achieve this, we call „The 2train standard setup“.

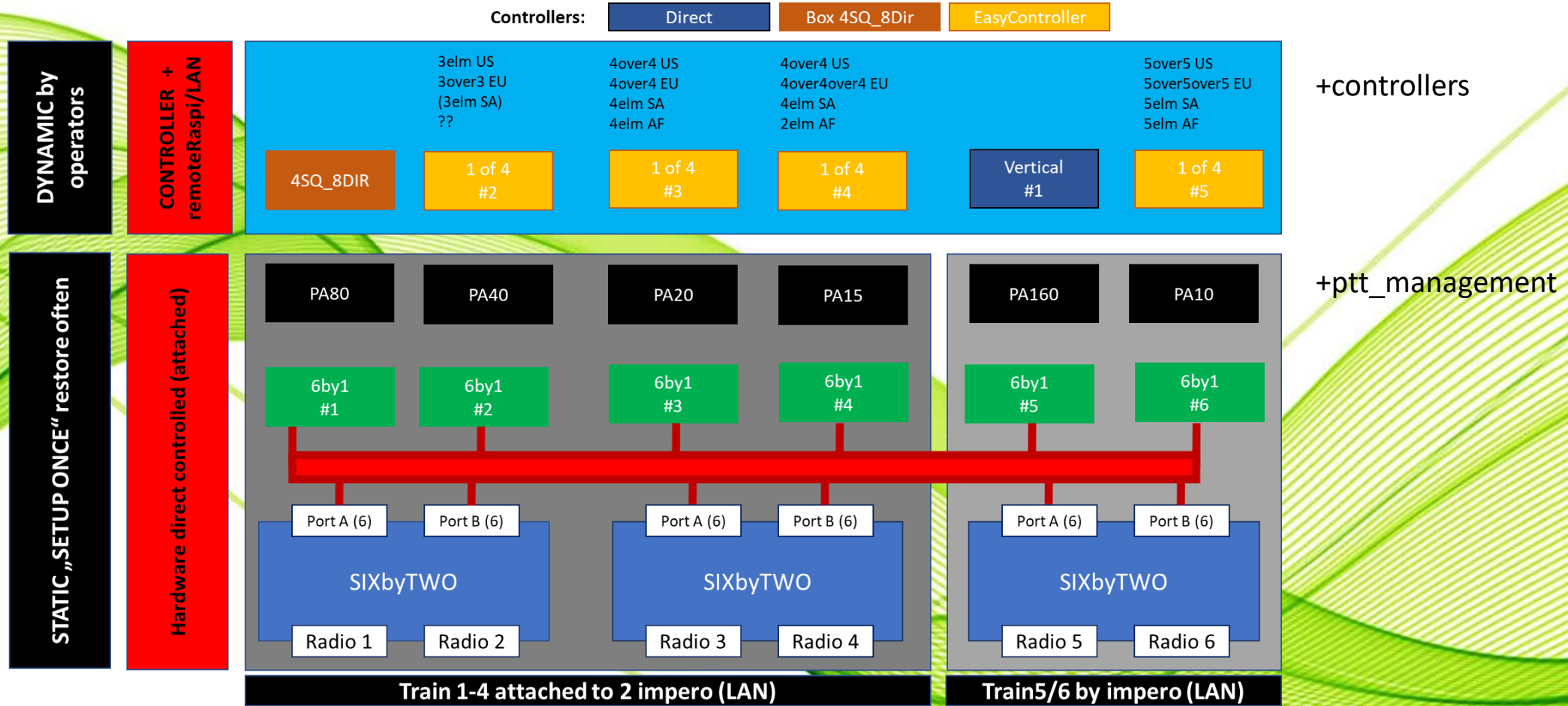
But also in this small setup, you have always at least 3 actors to switch:

- 2x a 2 by X (6/8/10 from hamparts.shop)
- ptt management (e.g. OLIP16)

**Time for automation!**



## Are you a super big?



This setup contains more 250 relays! Want to config and change all this by hand?  
STORE & RECALL!

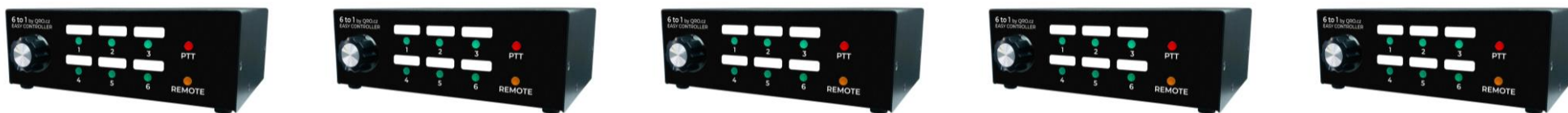


## What is a „controllers / station setup“?

Connecting the qro.cz controllers is easy and plug an play: Power it on, connect it to your shack network via lan cable. See the [„Your entry point“](#) to gain more information.

Each controller gets ist own IP address and you can speak „http“ or „ws (websockets)“ => see [remoteSwitchFramework](#)

So the setup at your station consists of one or many qro.cz-controllers. This is the stations infrastructure.



LAN / ETHERNET / WAN

## What is a controllers setup-configuration of a station?

All configuration are done using the file-system and a data format called [JSON](#). To introduce your controllers to the Super.Station.Automation, there is a (default/example) JSON-File called „controllers.json“ (in the dir „JSON“). This files needs to be edited first.

**Example:** 2 Controllers are in the network. 1 [Remotius](#) and 1 [OLIIP16](#)

```
[
  {
    "Name": "Power",
    "ContollerIp": "192.168.0.143",
    "Protocol": "http"
  },
  {
    "Name": "Remotius",
    "ContollerIp": "192.168.0.104",
    "Protocol": "ws"
  }
]
```

Give a controller a name

Add the controlles ip

The communication mode

Use „ws“ if possible

You will learn how to configure this later :p



## What is a „state“ of a controller?

The state of a controller is its set output-pins. The gro.cz controllers are using one (most controllers) or up to 4 (remotius) so called „banks“. Each bank represents 16 outputs (count starts with 0 for the first pin,...15).

Some controllers don't expose all pins, but under the hood, all is matched to a bank.

Since you have one to many controllers in your network, you can collect their states and store them as a file, e.g. JSON->statesWPX2024.json. You can have as many of them as you want – you can load and set the states persisted in the file to your station setup.

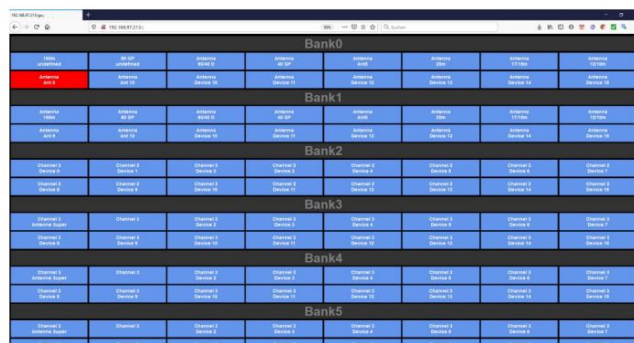
```
[
  {
    "Status": {
      "B0": 0,
      "LockStatus": false,
      "Interlock": false
    },
    "Mode": "http",
    "ip": "192.168.97.143"
  },
  {
    "Status": {
      "B0": 15,
      "B1": 4,
      "B2": 15,
      "B3": 257,
      "B4": 1048,
      "LockStatus": false,
      "Interlock": false
    },
    "Mode": "ws",
    "ip": "192.168.0.104"
  }
]
```

**Hint:** There is nothing to edit manually here.  
Just save and recall the file to the controllers and you are done!

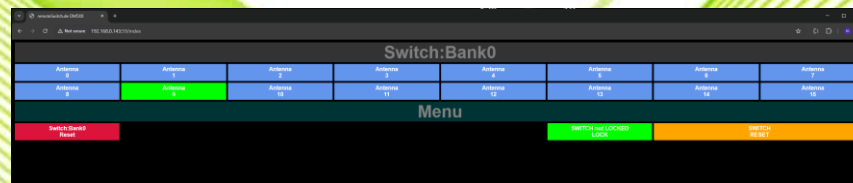
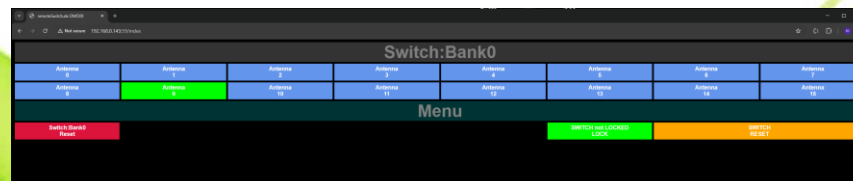
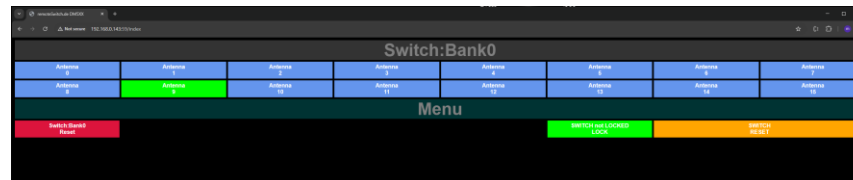
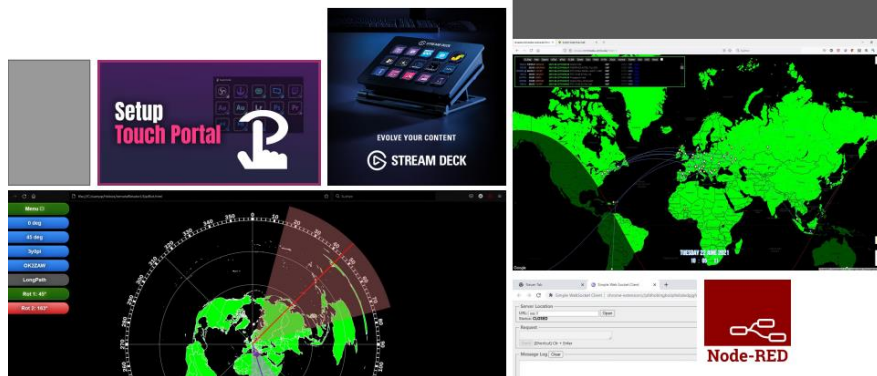


## What is the default „dashboard“ of a controller?

Using defaultUI



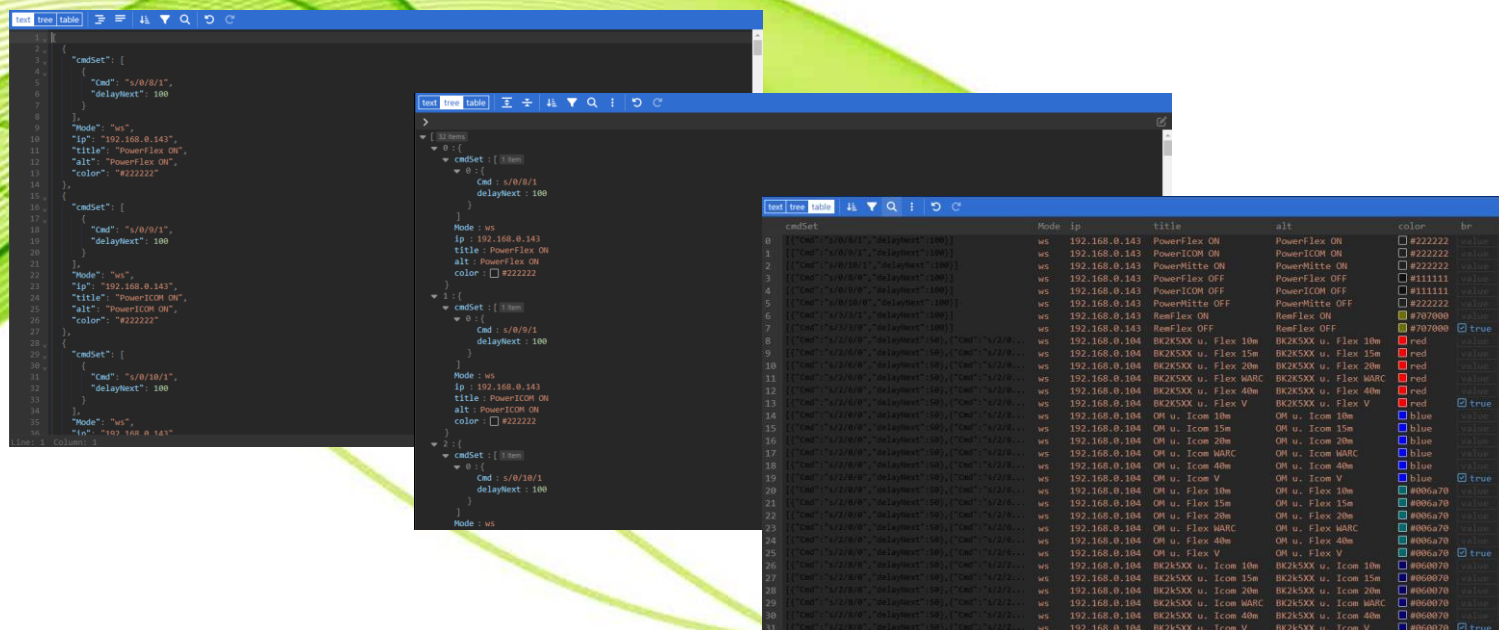
Using Web API



At the end, all controllers have their own dashboard (aka default UI) for many or just one bank. Orchestrating different dashboards needs some clicks. And that's where the **Super.Station.Automation** takes over!  
[Learn more about our UIs and APIs](#)

## Editing configurations: JSON Editor

For editing json files, the data format of the internet, we added a JSON Editor to the web application. Choos the way you like to edit your files: **Text**, **Tree**, **Table** or Table view

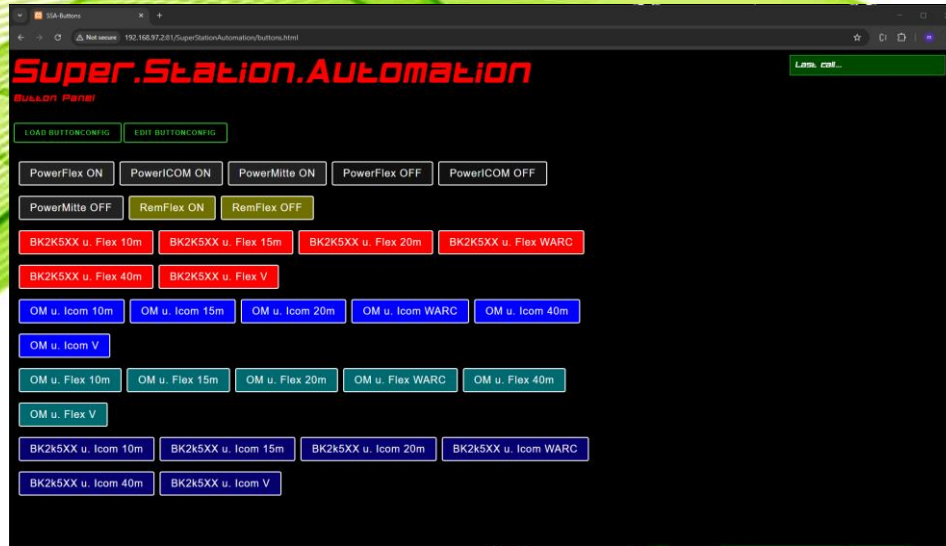


You can learn and get familiar with it trying the demo online version of the editor: <https://jsoneditoronline.org/>

**Hint:** Whatever file you edit, make sure you have a backup! So if you made a mistake, you always can go back to a working versions :P

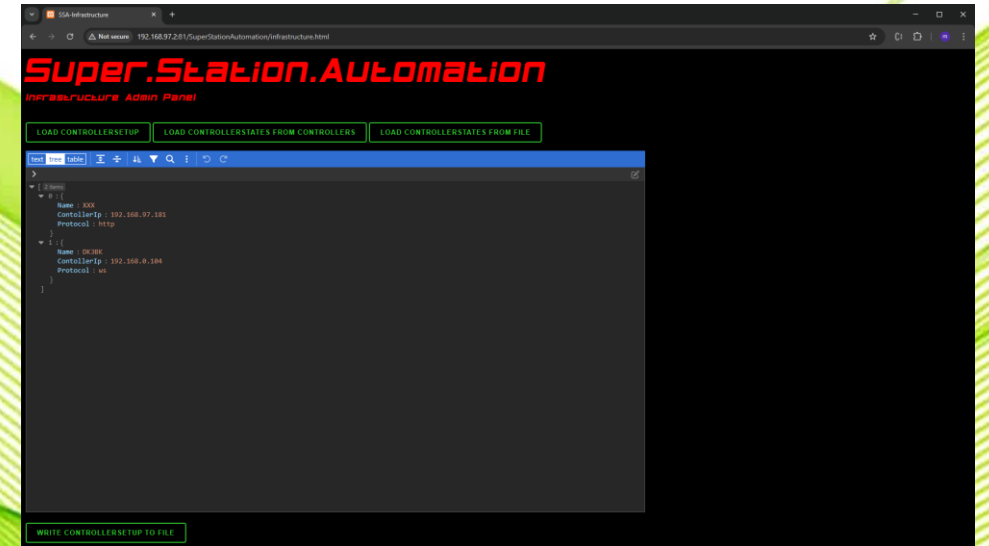


1



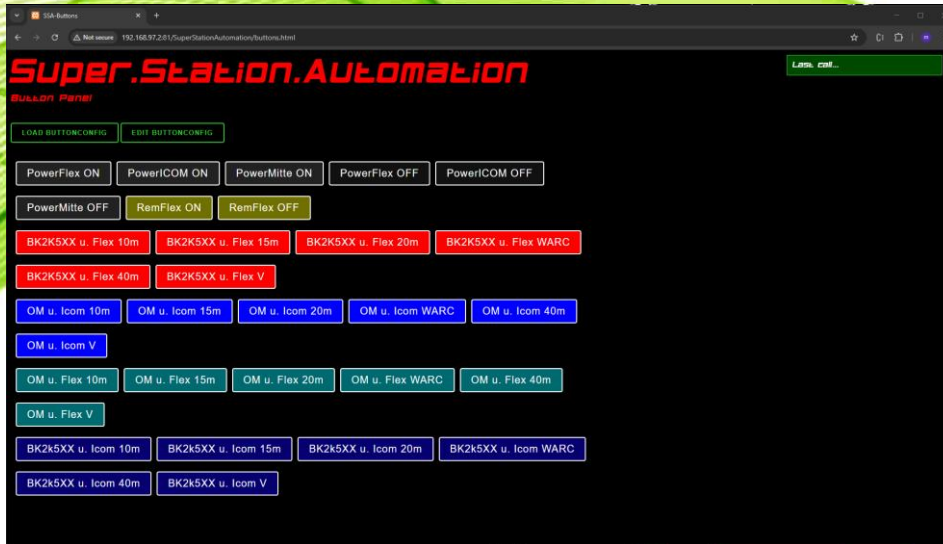
Button Panel

2



Infrastructure Admin Panel

1



## Button Panel

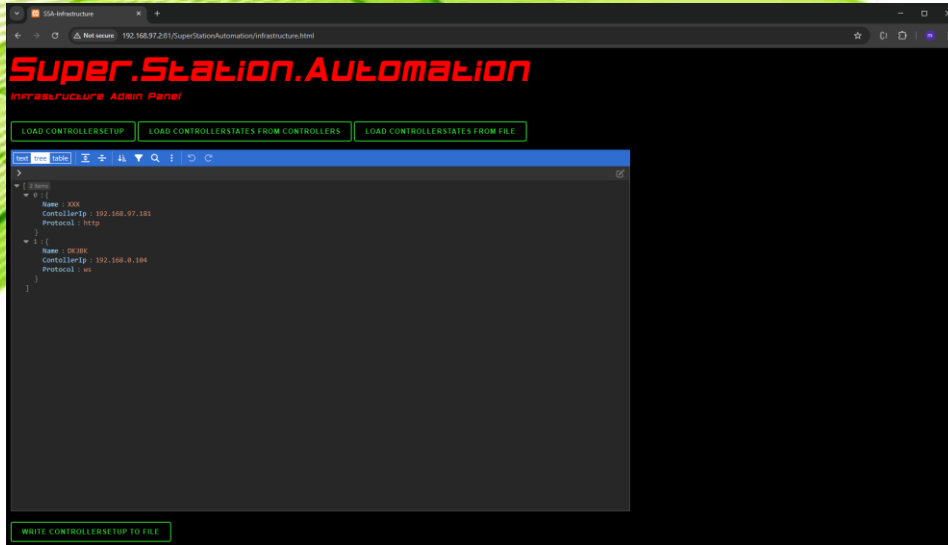
### What can i do?

- Load / Edit / Save a button-configuration (json)
- Click a desired button and execute it's command set (fire'n forget):  
=> Delayed execution of different controller commands aka chained

**Hint:** Create different dashboards for different use cases!



2



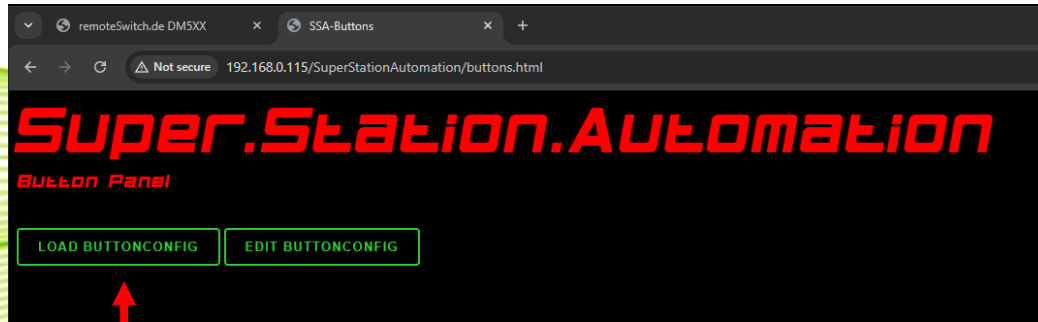
## Infrastructure Admin Panel

### What can i do?

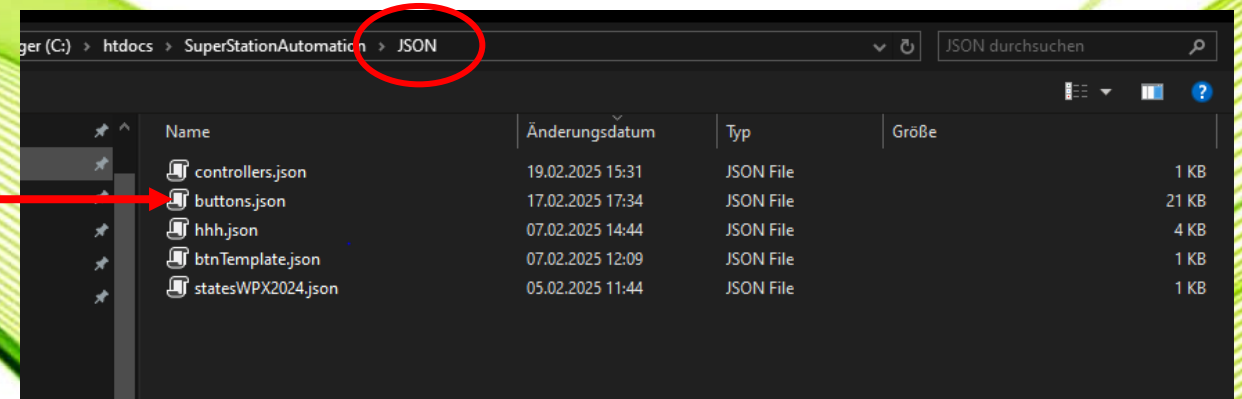
- Load / Edit / Save a controllersetup-configuration (json)
- Load/save current states of those controllers from/to the defined controllers
- Load/save states of those controllers from/to a file

**Hint:** Create different config files for different use cases!

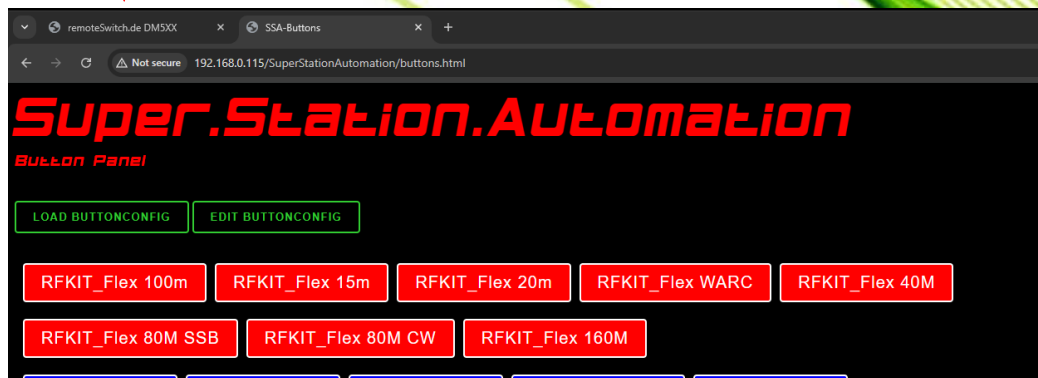
### Button Panel: Load a config (\*)



Load a button config



\*Load from local disc. Server see later

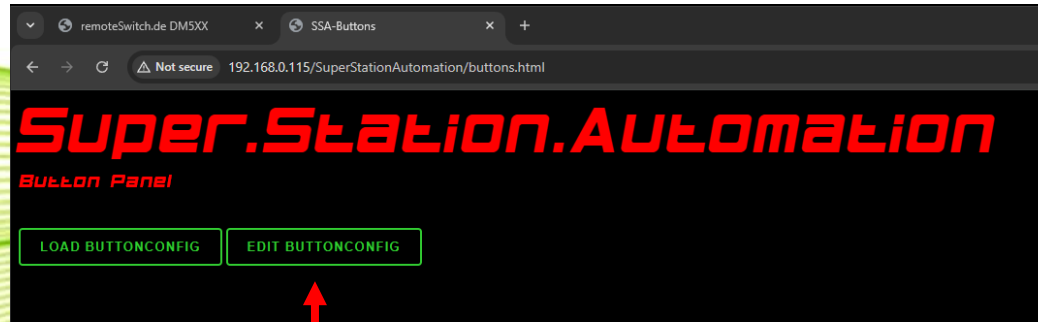




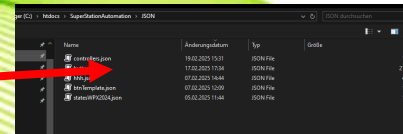
## Button Panel

1

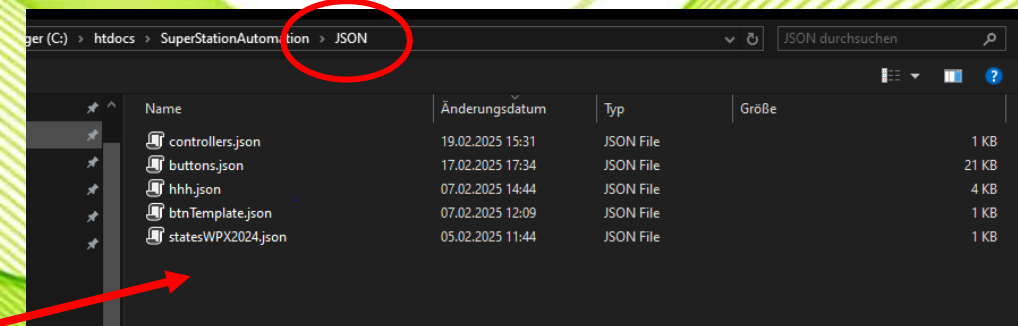
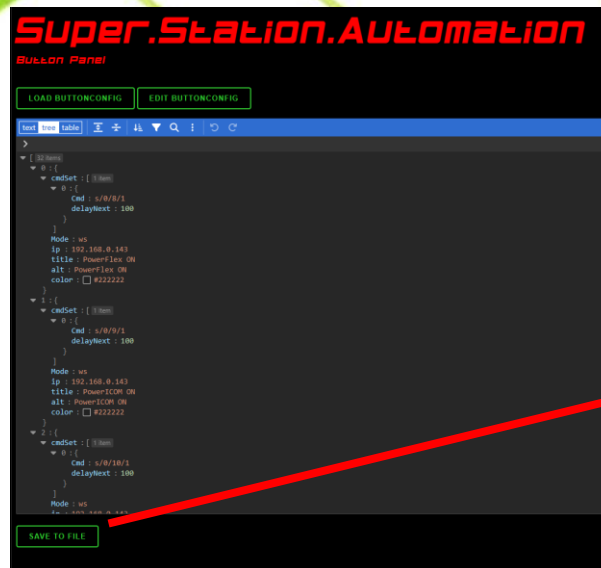
### Button Panel: Edit a config (\*)



Edit a button config



\*Load from local disc. Server see later



Save the current content of the editor:

- to an existing files (overwrite)
- to a new file (choose a useful file name)

Hint: Use Explorer for delete and rename!

## Buttons and command(chains)?

A button has a Mode (1), ip (2), title (3), alt text (4), a color (5) and a br/break (6) in its common definition. The commands are chained in command set (X) with a Cmd (7), a delayed exec.in ms (8) and a different Ip (9) and/or mode (10) if this specific command shd go to a different controller. A list of button objects creates your button panel.

```

90  "alt": "RemFlex ON",
91  "color": "#707000"
92  },
93  {
94    "cmdSet": [ (X)
95      {
96        "Cmd": "s/3/3/0",
97        "delayNext": 100
98      },
99      {
100       "Cmd": "set0/0/1", (7)
101       "delayNext": 100, (8)
102       "dip": "192.168.97.181", (9)
103       "mode": "http" (10)
104     },
105     {
106       "Cmd": "s/0/8/1",
107       "delayNext": 100,
108       "dip": "192.168.97.181",
109       "mode": "ws"
110     },
111     {
112       "Cmd": "s/0/0/1",
113       "delayNext": 100,
114       "dip": "192.168.0.104",
115       "mode": "ws"
116     }
117   ],
118   "Mode": "ws", (1)
119   "ip": "192.168.0.143", (2)
120   "title": "RemFlex OFF", (3)
121   "alt": "RemFlex OFF", (4)
122   "color": "#707000", (5)
123   "br": true (6)
124 },
125 {

```

**Hint: Make yourself familiar with the JSON-Notation.**  
Check the meaning of [ ] { } and ,

**Use the provided JSON-Editor or other online Editors.**  
**Use Textview or experiment with tree and table view.**  
**Use the view u like most.**  
**Dont forget to backup files :P**

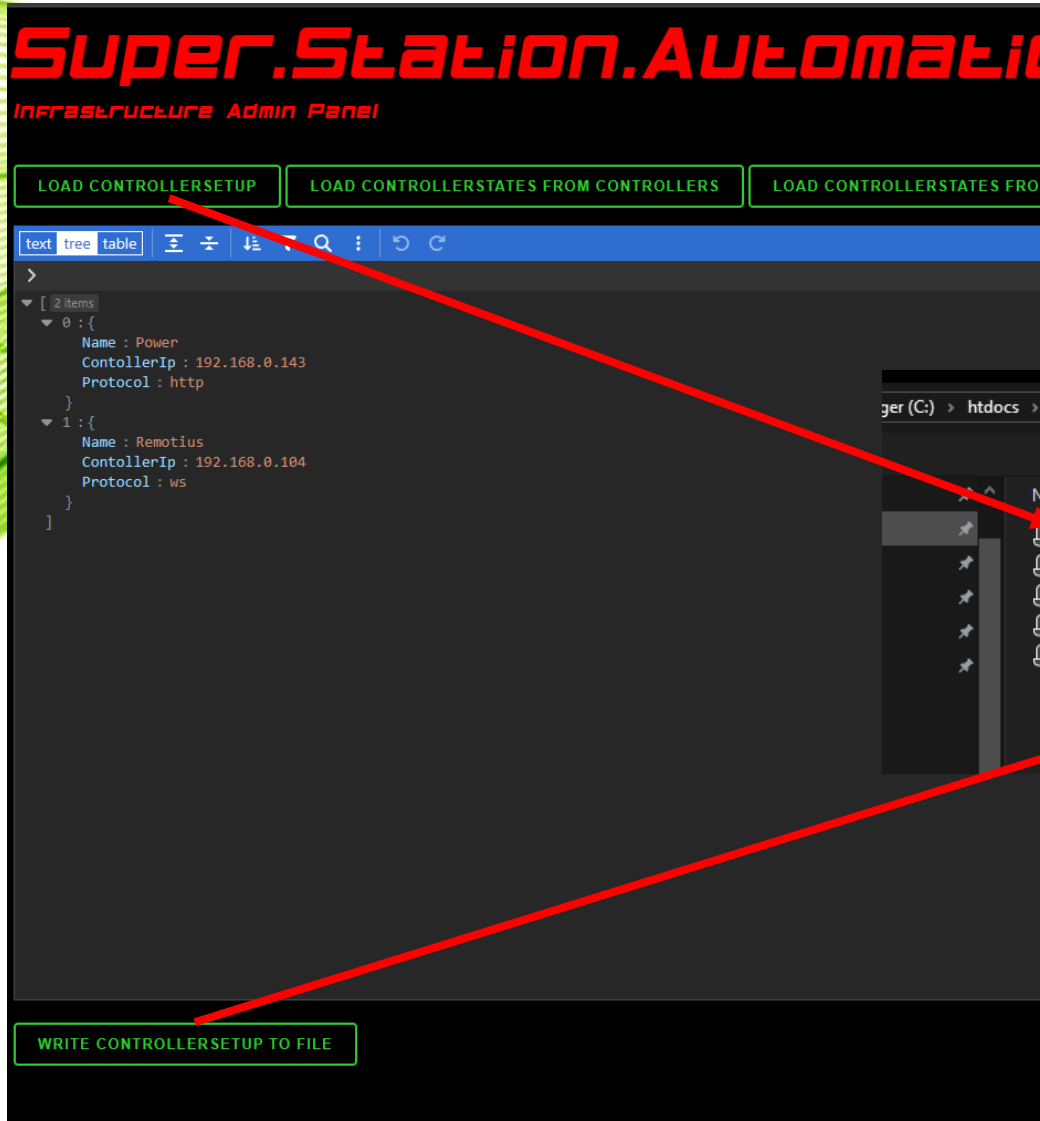
**See how to use the API/Cmds:**  
[remoteSwitchFramework slide 6](#)



## Infrastructure Admin Panel

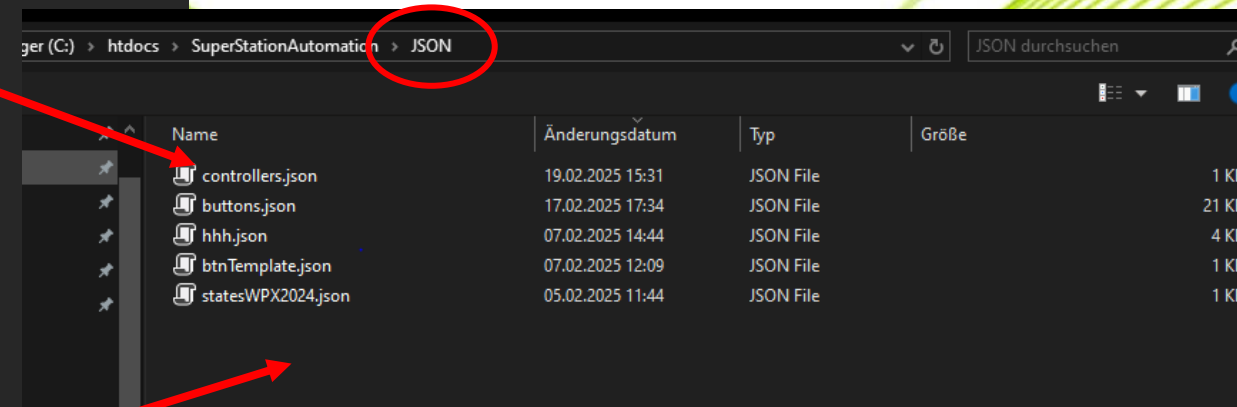
2

Load/Edit/Write a controllers setup config (\*)



**1. step is always load controllers!**

\*Load from local storage only!



**Save the current content of the editor:**

- to an existing files (overwrite)
- to a new file (choose a useful file name)

**Hint:** Use Explorer for delete and rename!

## Infrastructure Admin Panel

2

Load/Write controller states (\*)

**Super.Station.Automation**  
Infrastructure Admin Panel

LOAD CONTROLLERSETUP | LOAD CONTROLLERSTATES FROM CONTROLLERS | LOAD CONTROLLERSTATES FROM FILE

text | tree | table | [Icons]

```
1  [
2  {
3    "Status": {
4      "B0": 512,
5      "LockStatus": false,
6      "Interlock": false
7    },
8    "Mode": "http",
9    "ip": "192.168.0.143"
10  }
11  {
12    "Status": {
13      "B1": 1024,
14      "B2": 1024,
15      "B3": 1024,
16      "B4": 84,
17      "LockStatus": false,
18      "Interlock": false
19    },
20    "Mode": "ws",
21    "ip": "192.168.0.104"
22  }
23  }
24 ]
```

Line: 1 Column: 1

WRITE CONTROLLERSETUP TO FILE | WRITE CONTROLLERSTATES BACK TO CONTROLLERS | WRITE CONTROLLERSTATES TO A FILE

Loading the states from the controllers set up in the config via the network!

Loading the states from a controllers states file!

## 2. step is load/write states!

\*Load from local storage only!

**Write the current content of the editor:**

- to to the contollers via network. All controllers must be online and spec. (see 1. step)
- to a new controllers state file (choose a useful file name)

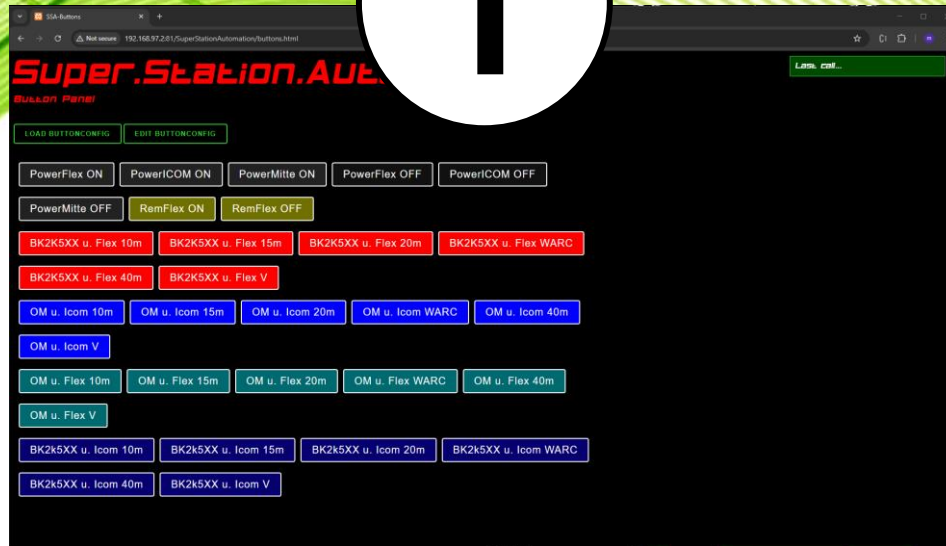
**Hint:** Take care that controllers setup file and controllers state file match 1:1 (check IPs!). Normally you dont edit something. You just load the states and write them back to have your station set to a specific state/configuration.



GITHUB: <https://github.com/dm5xx/SuperStationAutomation>

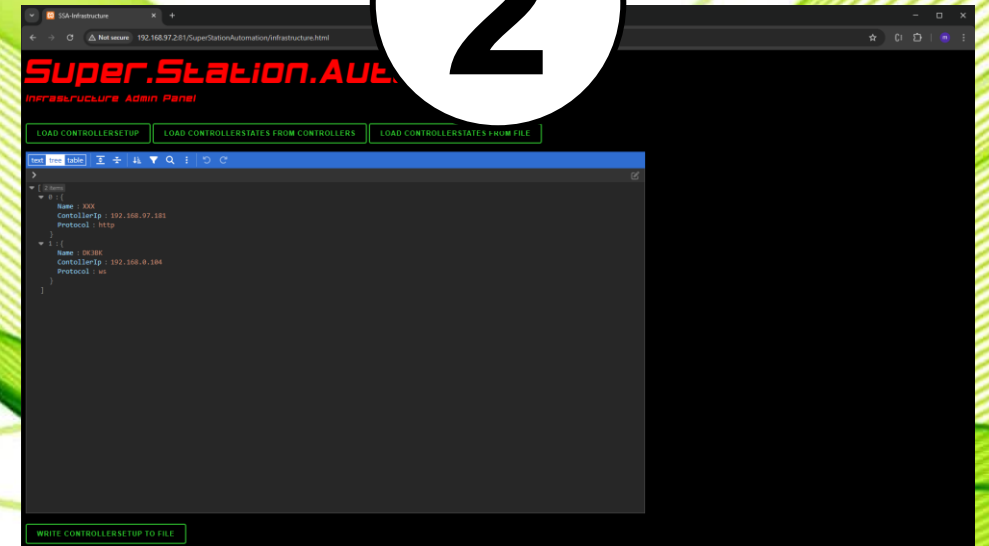
URLs/Pagenames for local use:

1



buttons.html

2



infrastructure.html

### URLs/Pagenames for server use:

1

<http://<YourServersIP>/SuperStationAutomation/creator.php?fn=buttons&jn=buttons>

Name of the buttons-config. Here: „buttons“ calls button.json.  
Change this to ur needs. No explicit loading required!

2

Since the infrastructure is always some more generic, there is no server side save and recall implemented. Use infrastructure.html and you local NAS/storage.

**Hint:** Server need PHP to run the software. Be aware that the  
JSON dir and file should have permission 755



## What does the example cmdSet do?

```

90  "alt": "RemFlex ON",
91  "color": "#707000"
92  },
93  {
94    "cmdSet": [
95      {
96        "Cmd": "s/3/3/0",
97        "delayNext": 100
98      },
99      {
100       "Cmd": "set0/0/1",
101       "delayNext": 100,
102       "dip": "192.168.97.181",
103       "mode": "http"
104     },
105     {
106       "Cmd": "s/0/8/1",
107       "delayNext": 100,
108       "dip": "192.168.97.181",
109       "mode": "ws"
110     },
111     {
112       "Cmd": "s/0/0/1",
113       "delayNext": 100,
114       "dip": "192.168.0.104",
115       "mode": "ws"
116     }
117   ],
118   "Mode": "ws",
119   "ip": "192.168.0.143",
120   "title": "RemFlex OFF",
121   "alt": "RemFlex OFF",
122   "color": "#707000",
123   "br": true
124 },
125 ]

```

CmdSet belongs to default 192.168.0.142 using default ws protocol. Btn title is „RemFlex OFF“. Btn color is a deep gray. This Btn is the last one in a row, so „br“ is set to true to make a line break

The CmdSet has 4 chained commands.

- (1) Use defaults, Set bank „3“, pin 3 to „0“ (off). Delay 100ms.
- (2) Using mode http, set bank 0, pin 0 to „1“ (on) at the controller with the IP 192.168.97.181. Delay 100ms.
- (3) Using mode ws, set bank 0, pin 8 to „1“ (on) at the controller with the IP 192.168.97.181. Delay 100ms.
- (4) Using mode ws, set bank 0, pin 0 to „1“ (on) at the controller with the IP 192.168.0.104. Delay 100ms.

Hint: If you switch items that belong together for the same context, perform the „OFF“ always bevor the „ON“. Becareful what you do! Dont kill your TRX! :P

Need help? Ask for our setup service!

# Special setup service!

We offer a special setup and configuration service for our customers:

- Configuring groups and locks on your controller (custom firmware)
- Special Updates and configured firmware cut to your needs

And:

- Initial setup of Super.Station.Automation Software with controllers and buttons
- Install on your web server
- Videocalls and education on the software
- Custom Super Station Development

**Ask mike DM5XX – [dm5xx@gmx.de](mailto:dm5xx@gmx.de)**