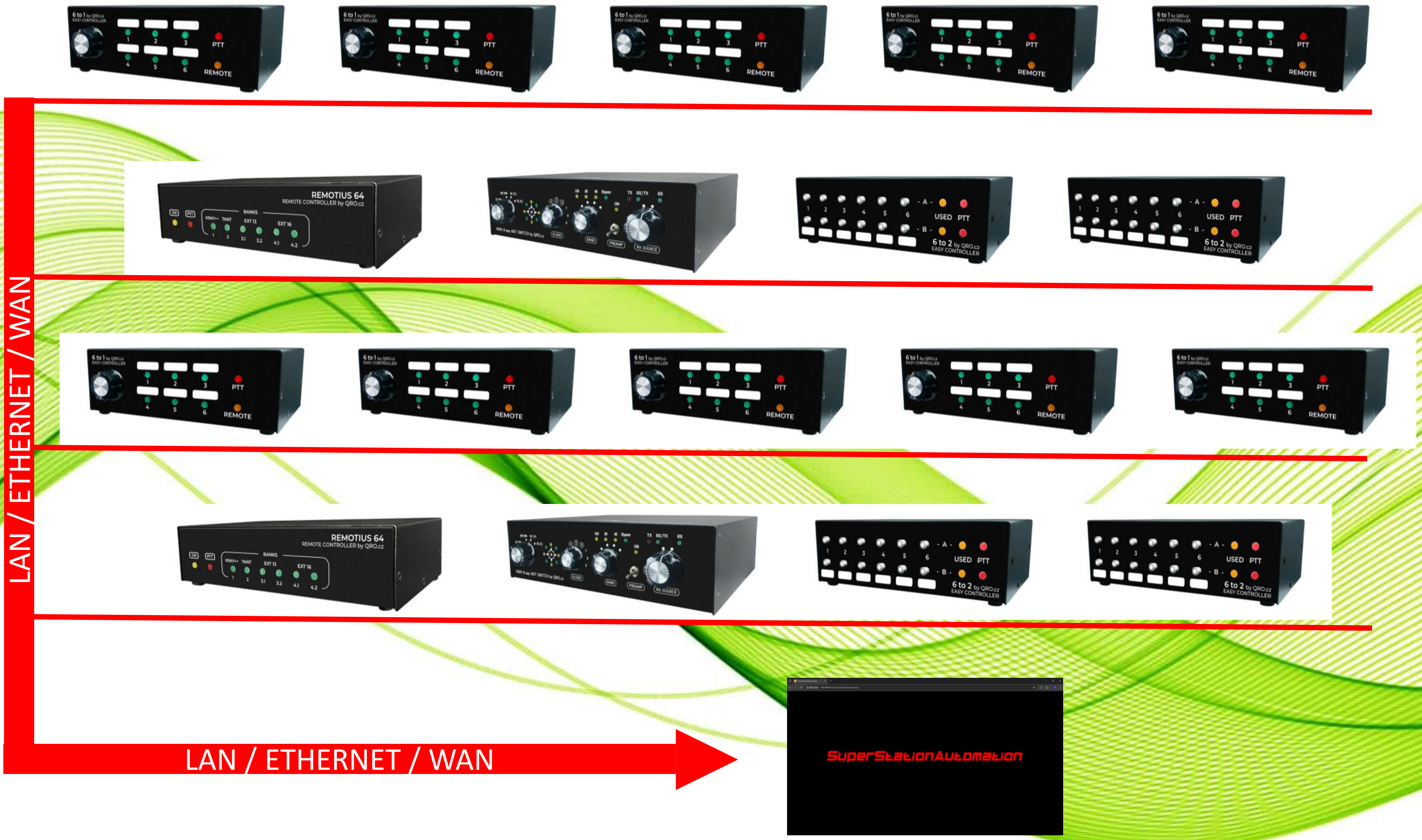
Super.Station.Automation

XX



6 to 1 by QRO.cz EASY CONTROLLER
PTT
REMOTE
REMOTIUS 64 REMOTE CONTROLLER by QRO.cz
6 to 2 by QRO.cz EASY CONTROLLER
USED
LAN / ETHERNET / WAN
LAN / ETHERNET / WAN
SuperStationAutomation

Super.Station.Automation

XX

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 / 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:

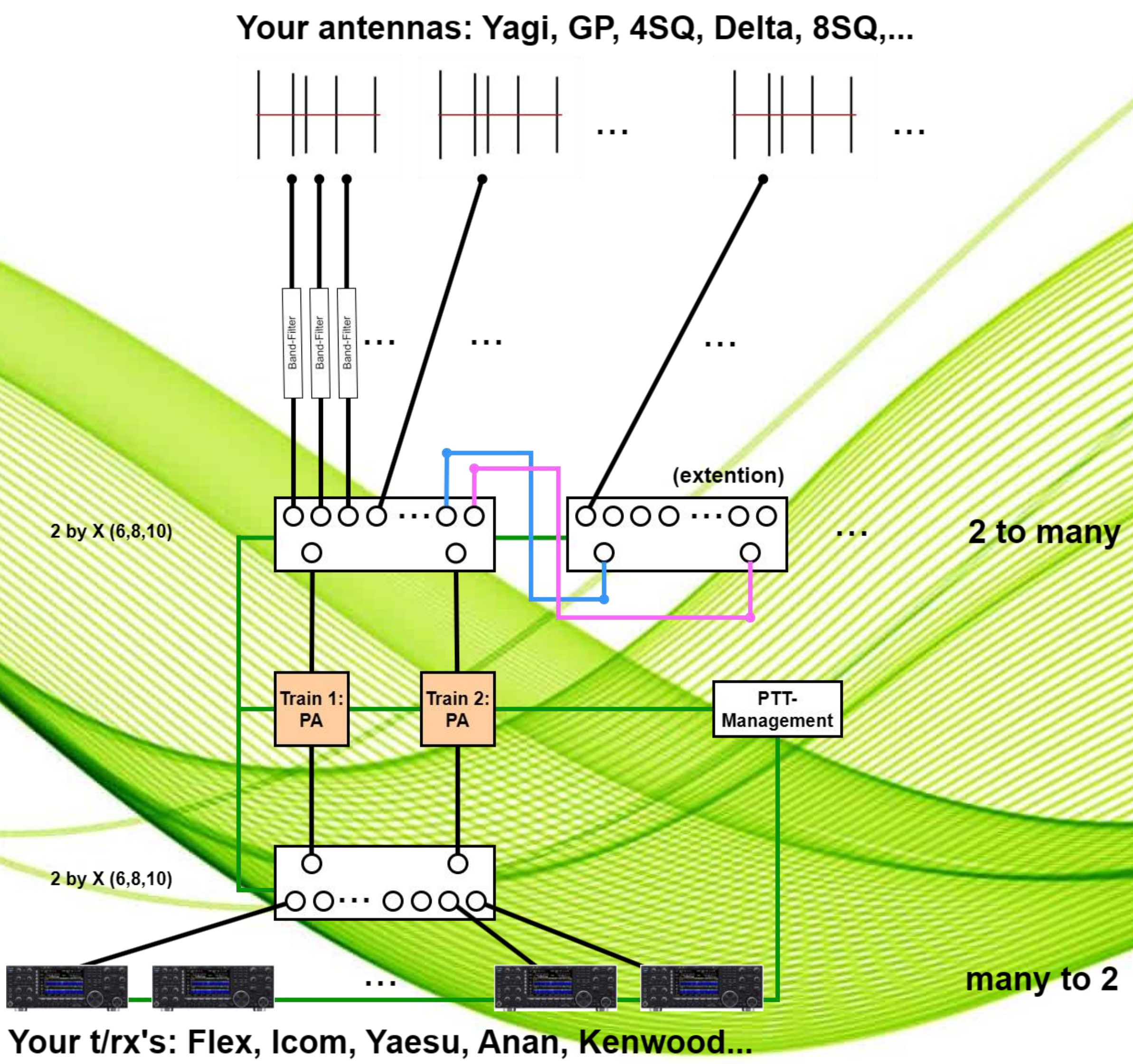
- many have more antennas
- many have even more ants per band
- many have more then 1 trx :p
- many want to have 2 signals on the air
- many have more than one amp
- doing contest in a team is fun

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

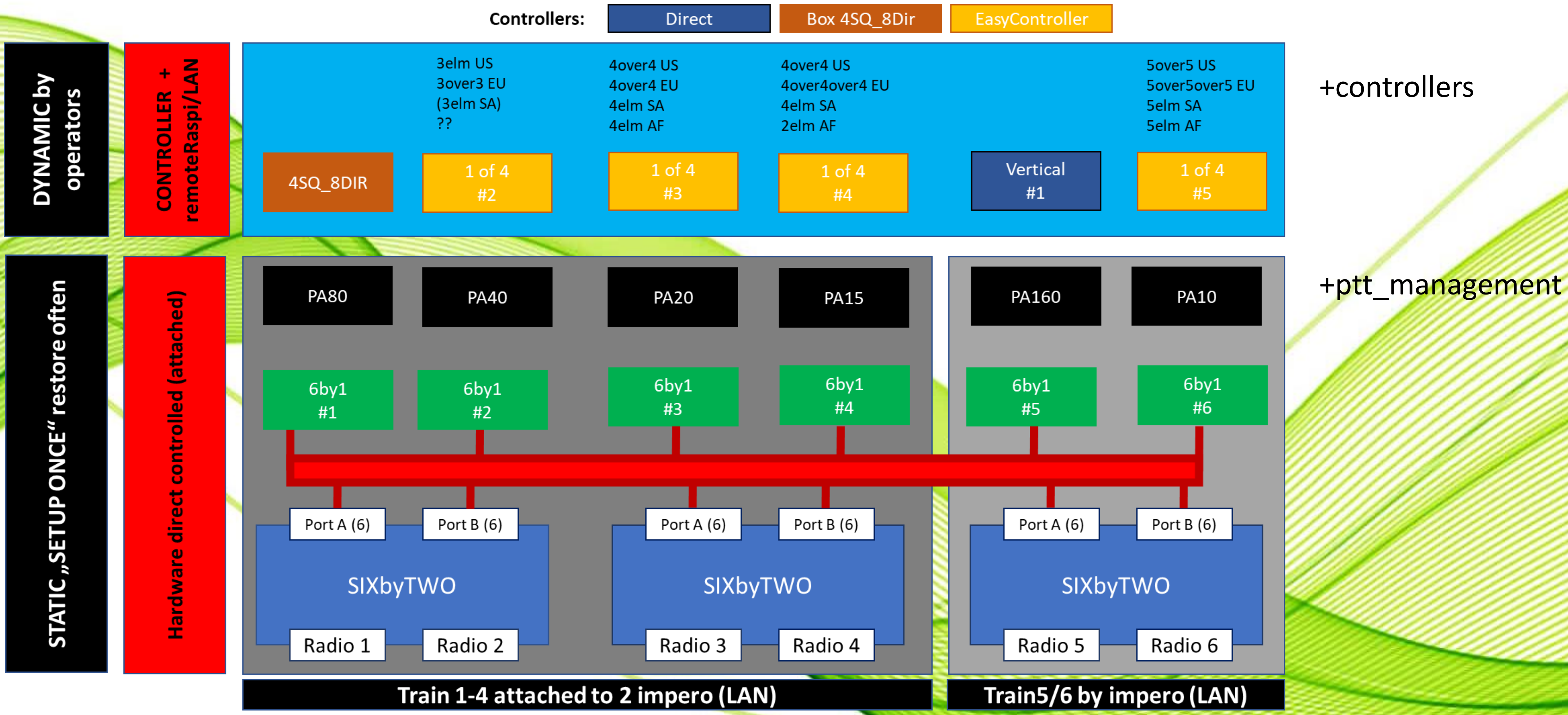
But also in this small setup, you have always at least 3 actors to switch:

- 2x a 2 by X (6/8/10 from hamparts.shop)
- ptt management (e.g. OLIIP16)

Time for automation!



Are you a super big?

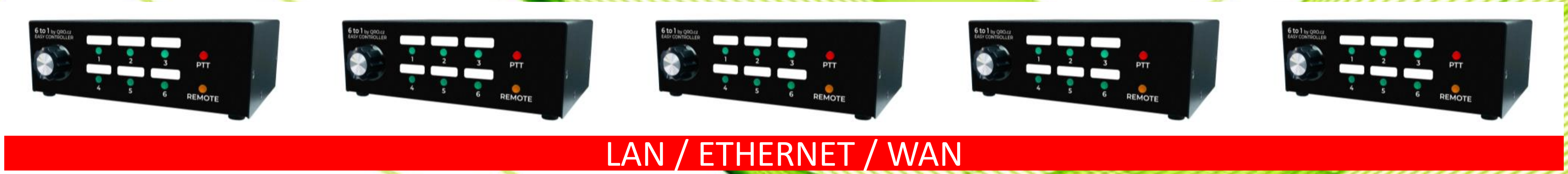


This setup contains more 250 relays! Want to config and change all this by hand?
STORE & RECALL!

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

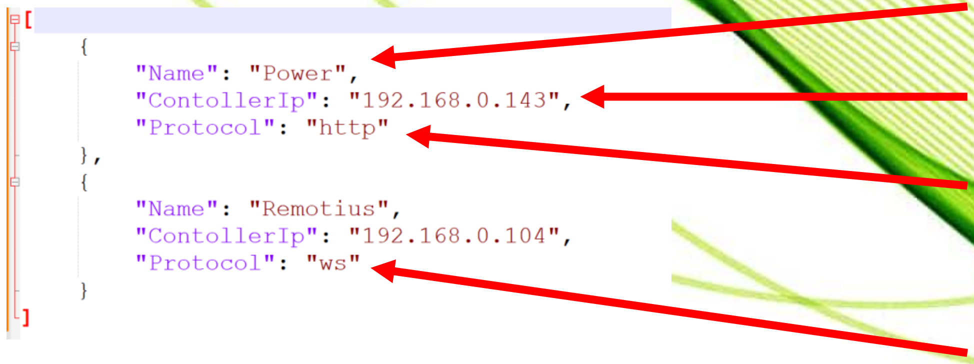
So the setup at your station consists of one or many qro.cz-controllers. This is the stations infrastructure.



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

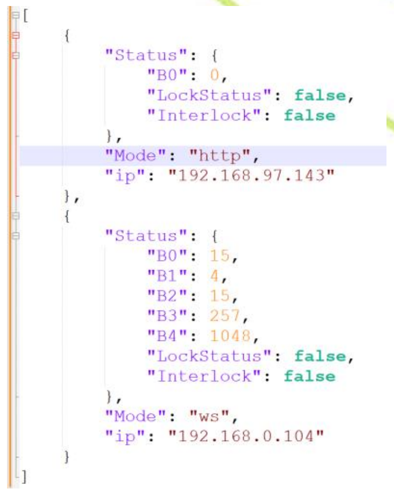
Super.Station.Automation

XX

What is a „state“ of a controller?

The state of a controller is ist set output-pins. The qro.cz controller 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 dont 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 persited in the file to you station setup.

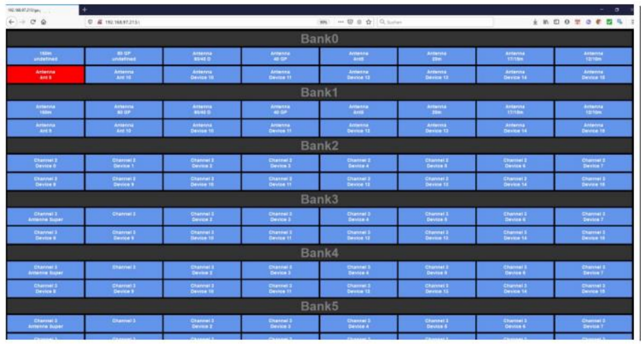


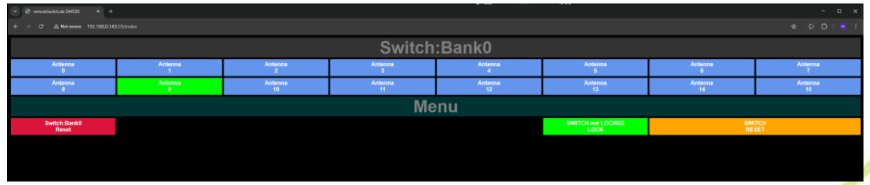
```
[
    {
        "Status": {
            "B0": 0,
            "LockStatus": false,
            "Interlock": false
        },
        "Mode": "http",
        "ip": "192.168.97.143"
    },
    {
        "Status": {
            "B0": 15,
            "B1": 4,
            "B2": 15,
            "B3": 257,
            "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!

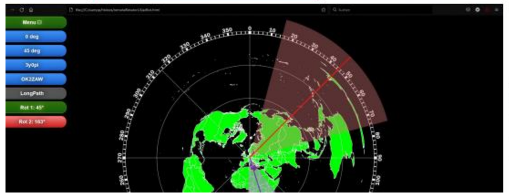
What is the default „dashboard“ of a controller?

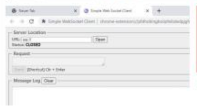
Using defaultUI





Using Web API



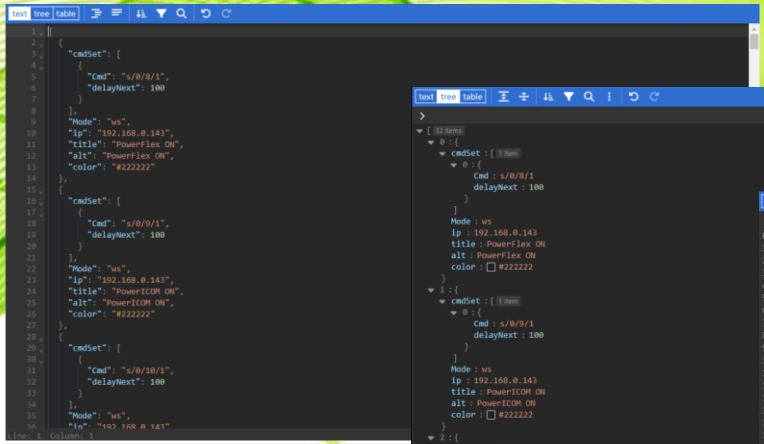


Node-RED

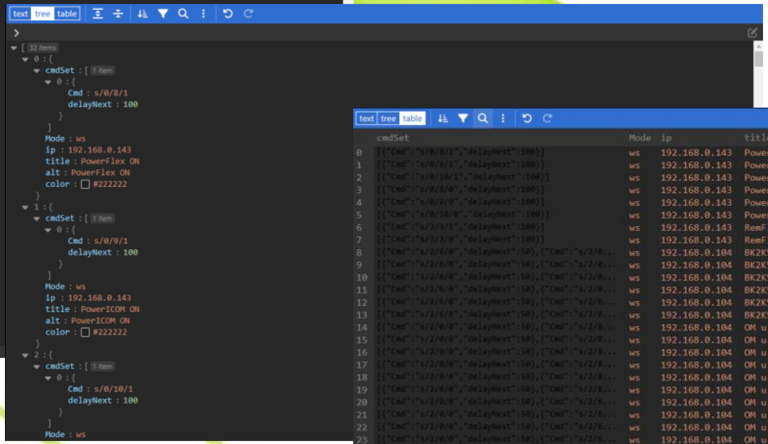
At the end, all controllers have their own dashboard (aka default UI) for many or just one bank. Orchistrating different dashboards needs some clicks. And thats where the **Super.Station.Automation** takes over!
Learn more about our UIs and APIs

Editing configurations: JSON Editor

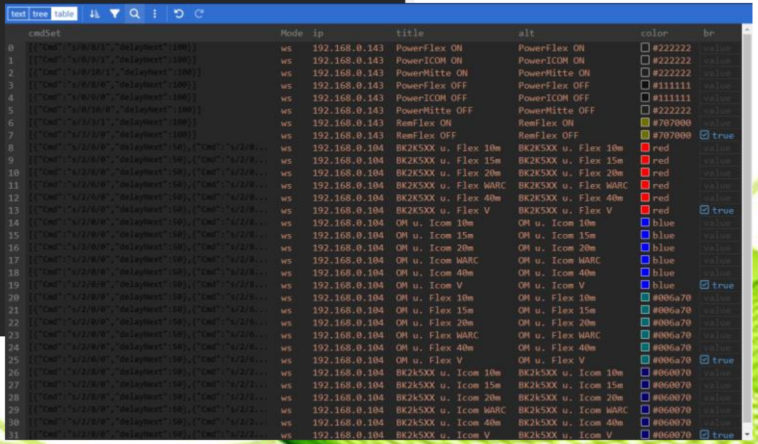
For editing json files, the data format of the internet, we added a JSON Editor to the web application. Choos the way you like to edit your files: **Text, Tree, Table** or Table view



```
[
  {
    "cmdSet": [
      {
        "Cmd": "s/0/8/1",
        "delayNext": 100
      }
    ],
    "Mode": "ws",
    "ip": "192.168.0.143",
    "title": "PowerFlex ON",
    "alt": "PowerFlex ON",
    "color": "#222222"
  },
  {
    "cmdSet": [
      {
        "Cmd": "s/0/9/1",
        "delayNext": 100
      }
    ],
    "Mode": "ws",
    "ip": "192.168.0.143",
    "title": "PowerICOM ON",
    "alt": "PowerICOM ON",
    "color": "#222222"
  },
  {
    "cmdSet": [
      {
        "Cmd": "s/0/10/1",
        "delayNext": 100
      }
    ],
    "Mode": "ws",
    "ip": "192.168.0.143",
Line: 1  Column: 1
```



```
[ 32 items
  0 :{
    cmdSet : [ 1 item
      0 :{
          Cmd : s/0/8/1
          delayNext : 100
        }
      ]
    Mode : ws
    ip : 192.168.0.143
    title : PowerFlex ON
    alt : PowerFlex ON
    color : □ #222222
  }
  1 :{
    cmdSet : [ 1 item
      0 :{
          Cmd : s/0/9/1
          delayNext : 100
        }
      ]
    Mode : ws
    ip : 192.168.0.143
    title : PowerICOM ON
    alt : PowerICOM ON
    color : □ #222222
  }
  2 :{
    cmdSet : [ 1 item
      0 :{
          Cmd : s/0/10/1
          delayNext : 100
        }
      ]
    Mode : ws
```



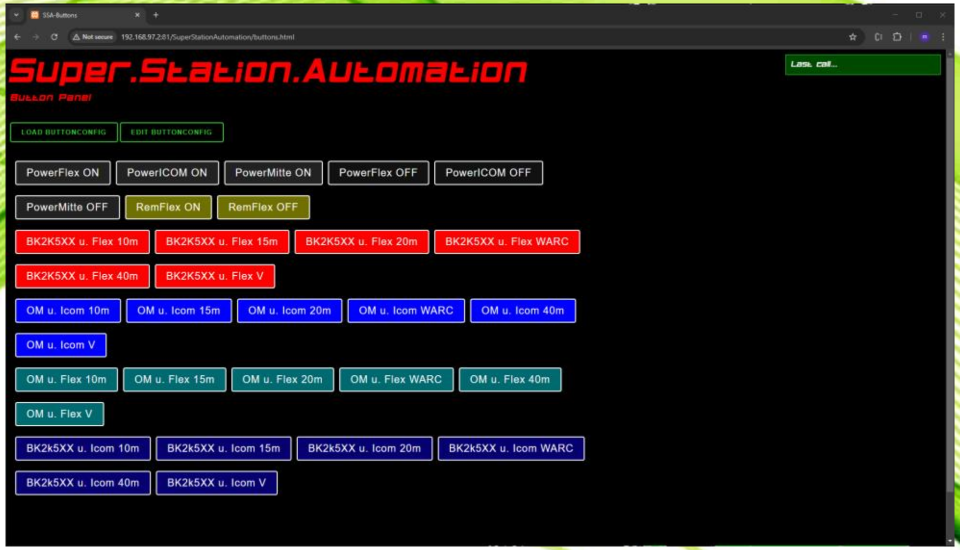
	cmdSet	Mode	ip	title	alt	color	br
0	[{"Cmd":"s/0/8/1","delayNext":100}]	ws	192.168.0.143	PowerFlex ON	PowerFlex ON	□ #222222	value
1	[{"Cmd":"s/0/9/1","delayNext":100}]	ws	192.168.0.143	PowerICOM ON	PowerICOM ON	□ #222222	value
2	[{"Cmd":"s/0/10/1","delayNext":100}]	ws	192.168.0.143	PowerMitte ON	PowerMitte ON	□ #222222	value
3	[{"Cmd":"s/0/8/0","delayNext":100}]	ws	192.168.0.143	PowerFlex OFF	PowerFlex OFF	■ #111111	value
4	[{"Cmd":"s/0/9/0","delayNext":100}]	ws	192.168.0.143	PowerICOM OFF	PowerICOM OFF	■ #111111	value
5	[{"Cmd":"s/0/10/0","delayNext":100}]	ws	192.168.0.143	PowerMitte OFF	PowerMitte OFF	□ #222222	value
6	[{"Cmd":"s/3/3/1","delayNext":100}]	ws	192.168.0.143	RemFlex ON	RemFlex ON	■ #707000	value
7	[{"Cmd":"s/3/3/0","delayNext":100}]	ws	192.168.0.143	RemFlex OFF	RemFlex OFF	■ #707000	☑ true
8	[{"Cmd":"s/2/0/0","delayNext":50},{"Cmd":"s/2/0...	ws	192.168.0.104	BK2K5XX u. Flex 10m	BK2K5XX u. Flex 10m	🟥 red	value
9	[{"Cmd":"s/2/6/0","delayNext":50},{"Cmd":"s/2/0...	ws	192.168.0.104	BK2K5XX u. Flex 15m	BK2K5XX u. Flex 15m	🟥 red	value
10	[{"Cmd":"s/2/6/0","delayNext":50},{"Cmd":"s/2/0...	ws	192.168.0.104	BK2K5XX u. Flex 20m	BK2K5XX u. Flex 20m	🟥 red	value
11	[{"Cmd":"s/2/6/0","delayNext":50},{"Cmd":"s/2/0...	ws	192.168.0.104	BK2K5XX u. Flex WARC	BK2K5XX u. Flex WARC	🟥 red	value
12	[{"Cmd":"s/2/6/0","delayNext":50},{"Cmd":"s/2/0...	ws	192.168.0.104	BK2K5XX u. Flex 40m	BK2K5XX u. Flex 40m	🟥 red	value
13	[{"Cmd":"s/2/6/0","delayNext":50},{"Cmd":"s/2/0...	ws	192.168.0.104	BK2K5XX u. Flex V	BK2K5XX u. Flex V	🟥 red	☑ true
14	[{"Cmd":"s/2/0/0","delayNext":50},{"Cmd":"s/2/8...	ws	192.168.0.104	OM u. Icom 10m	OM u. Icom 10m	🟦 blue	value
15	[{"Cmd":"s/2/0/0","delayNext":50},{"Cmd":"s/2/8...	ws	192.168.0.104	OM u. Icom 15m	OM u. Icom 15m	🟦 blue	value
16	[{"Cmd":"s/2/0/0","delayNext":50},{"Cmd":"s/2/8...	ws	192.168.0.104	OM u. Icom 20m	OM u. Icom 20m	🟦 blue	value
17	[{"Cmd":"s/2/0/0","delayNext":50},{"Cmd":"s/2/8...	ws	192.168.0.104	OM u. Icom WARC	OM u. Icom WARC	🟦 blue	value
18	[{"Cmd":"s/2/0/0","delayNext":50},{"Cmd":"s/2/8...	ws	192.168.0.104	OM u. Icom 40m	OM u. Icom 40m	🟦 blue	value
19	[{"Cmd":"s/2/0/0","delayNext":50},{"Cmd":"s/2/8...	ws	192.168.0.104	OM u. Icom V	OM u. Icom V	🟦 blue	☑ true
20	[{"Cmd":"s/2/0/0","delayNext":50},{"Cmd":"s/2/6...	ws	192.168.0.104	OM u. Flex 10m	OM u. Flex 10m	■ #006a70	value
21	[{"Cmd":"s/2/0/0","delayNext":50},{"Cmd":"s/2/6...	ws	192.168.0.104	OM u. Flex 15m	OM u. Flex 15m	■ #006a70	value
22	[{"Cmd":"s/2/0/0","delayNext":50},{"Cmd":"s/2/6...	ws	192.168.0.104	OM u. Flex 20m	OM u. Flex 20m	■ #006a70	value
23	[{"Cmd":"s/2/0/0","delayNext":50},{"Cmd":"s/2/6...	ws	192.168.0.104	OM u. Flex WARC	OM u. Flex WARC	■ #006a70	value
24	[{"Cmd":"s/2/0/0","delayNext":50},{"Cmd":"s/2/6...	ws	192.168.0.104	OM u. Flex 40m	OM u. Flex 40m	■ #006a70	value
25	[{"Cmd":"s/2/0/0","delayNext":50},{"Cmd":"s/2/6...	ws	192.168.0.104	OM u. Flex V	OM u. Flex V	■ #006a70	☑ true
26	[{"Cmd":"s/2/8/0","delayNext":50},{"Cmd":"s/2/2...	ws	192.168.0.104	BK2k5XX u. Icom 10m	BK2k5XX u. Icom 10m	■ #060070	value
27	[{"Cmd":"s/2/8/0","delayNext":50},{"Cmd":"s/2/2...	ws	192.168.0.104	BK2k5XX u. Icom 15m	BK2k5XX u. Icom 15m	■ #060070	value
28	[{"Cmd":"s/2/8/0","delayNext":50},{"Cmd":"s/2/2...	ws	192.168.0.104	BK2k5XX u. Icom 20m	BK2k5XX u. Icom 20m	■ #060070	value
29	[{"Cmd":"s/2/8/0","delayNext":50},{"Cmd":"s/2/2...	ws	192.168.0.104	BK2k5XX u. Icom WARC	BK2k5XX u. Icom WARC	■ #060070	value
30	[{"Cmd":"s/2/8/0","delayNext":50},{"Cmd":"s/2/2...	ws	192.168.0.104	BK2k5XX u. Icom 40m	BK2k5XX u. Icom 40m	■ #060070	value
31	[{"Cmd":"s/2/8/0","delayNext":50},{"Cmd":"s/2/2...	ws	192.168.0.104	BK2k5XX u. Icom V	BK2k5XX u. Icom V	■ #060070	☑ true

You can learn and get familar 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 verions :P

Super.Station.Automation

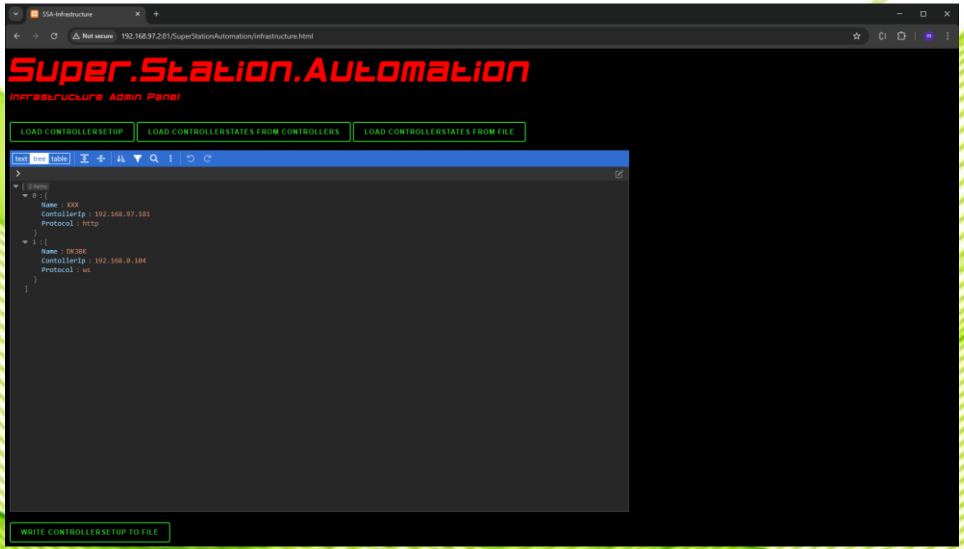
XX

①



Button Panel

②


SSA-Infrastructure
Not secure 192.168.97.281/SuperStationAutomation/infrastructure.html
Super.Station.Automation
Infrastructure Admin Panel
LOAD CONTROLLERSETUP
LOAD CONTROLLERSTATES FROM CONTROLLERS
LOAD CONTROLLERSTATES FROM FILE
text
tree
table
WRITE CONTROLLERSETUP TO FILE

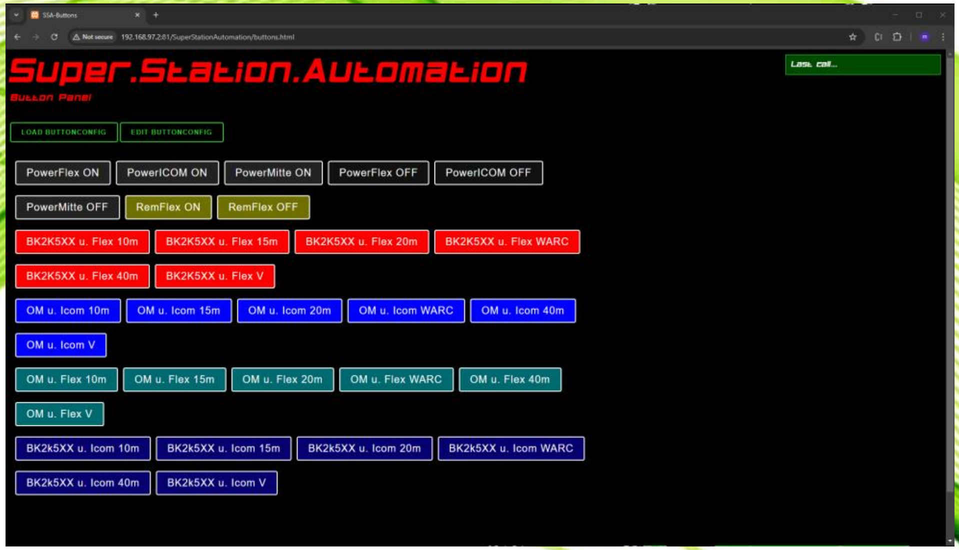
```
[ 2 items
  0 :{
    Name : XXX
    ContollerIp : 192.168.97.181
    Protocol : http
  }
  1 :{
    Name : DK3BK
    ContollerIp : 192.168.0.104
    Protocol : ws
  }
]
```

Infrastructure Admin Panel

Super.Station.Automation

XX

①



Button Panel

What can i do?

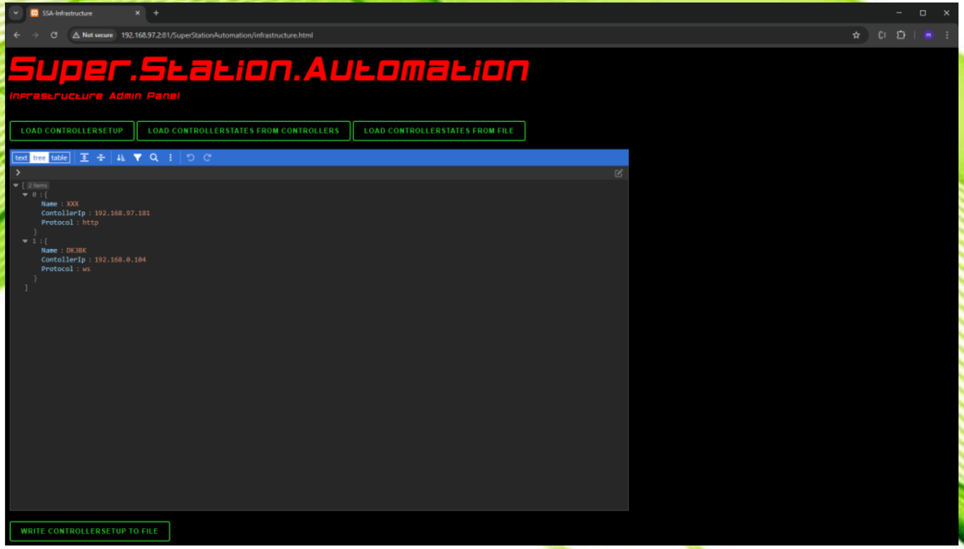
- Load / Edit / Save a button-configuration (json)
- Click a desired button and execute it's command set (fire'n forget):
  => Delayed execution of different controller commands aka chained

Hint: Create different dashboards for different use cases!

Super.Station.Automation

XX

2


SSA-Infrastructure
Not secure
192.168.97.2:81/SuperStationAutomation/infrastructure.html
Super.Station.Automation
Infrastructure Admin Panel
LOAD CONTROLLERSETUP
LOAD CONTROLLERSTATES FROM CONTROLLERS
LOAD CONTROLLERSTATES FROM FILE
text
tree
table
WRITE CONTROLLERSETUP TO FILE

```
[ 2 items
  0 :{
     Name : XXX
     ContollerIp : 192.168.97.181
     Protocol : http
  }
  1 :{
     Name : DK3BK
     ContollerIp : 192.168.0.104
     Protocol : ws
  }
]
```

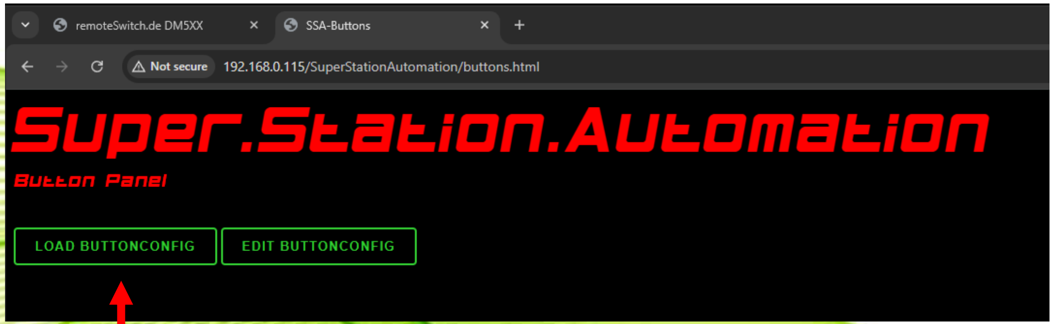
Infrastructure Admin Panel

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 (*)


remoteSwitch.de DM5XX | SSA-Buttons

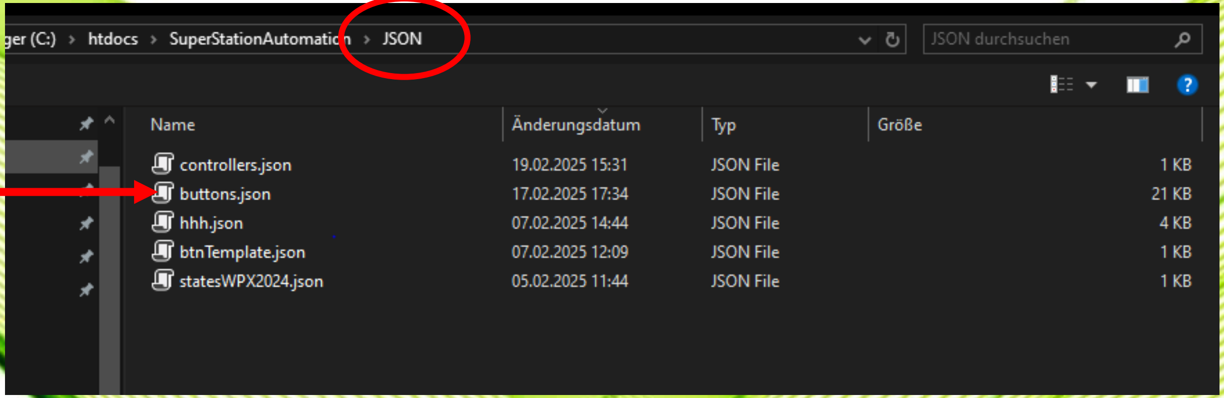
Not secure 192.168.0.115/SuperStationAutomation/buttons.html

Super.Station.Automation

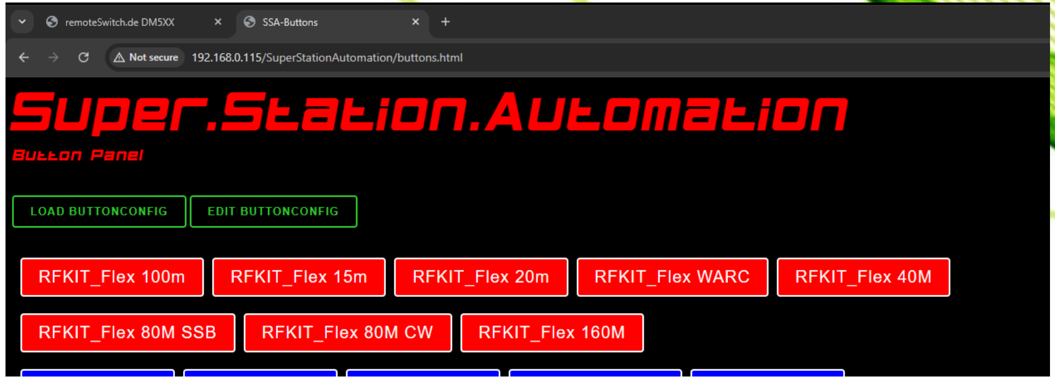
Button Panel

Load a button config

*Load from local disc. Server see later


ger (C:) › htdocs › SuperStationAutomation › JSON

Name	Änderungsdatum	Typ	Größe
controllers.json	19.02.2025 15:31	JSON File	1 KB
buttons.json	17.02.2025 17:34	JSON File	21 KB
hhh.json	07.02.2025 14:44	JSON File	4 KB
btnTemplate.json	07.02.2025 12:09	JSON File	1 KB
statesWPX2024.json	05.02.2025 11:44	JSON File	1 KB


remoteSwitch.de DM5XX | SSA-Buttons

Not secure 192.168.0.115/SuperStationAutomation/buttons.html

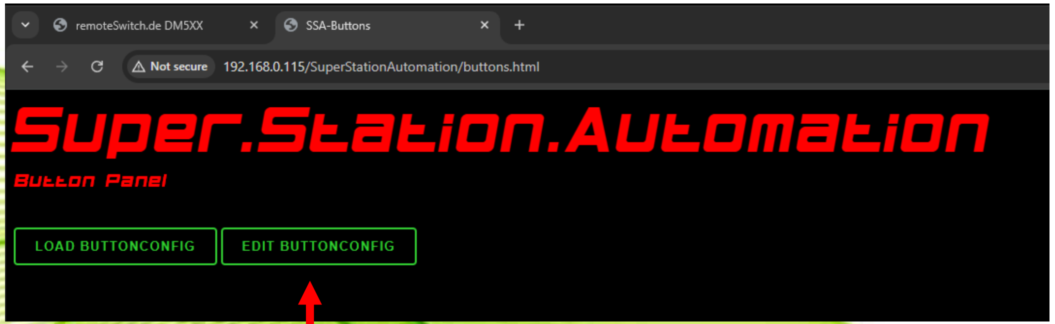
Super.Station.Automation

Button Panel

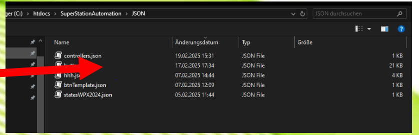
Super.Station.Automation XX

Button Panel ①

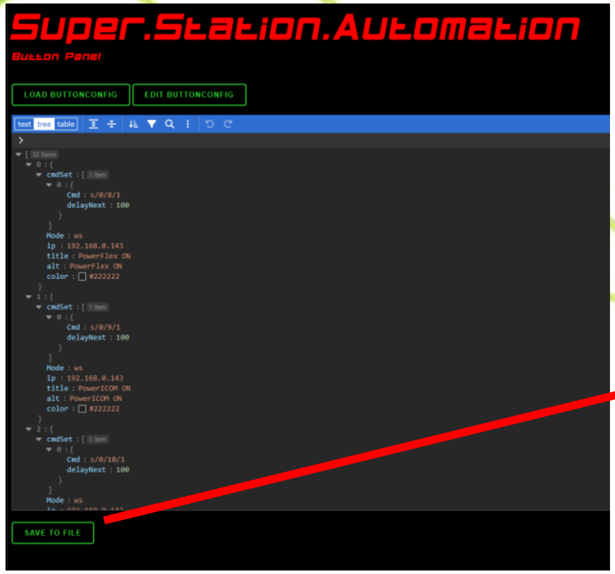
Button Panel: Edit a config (*)



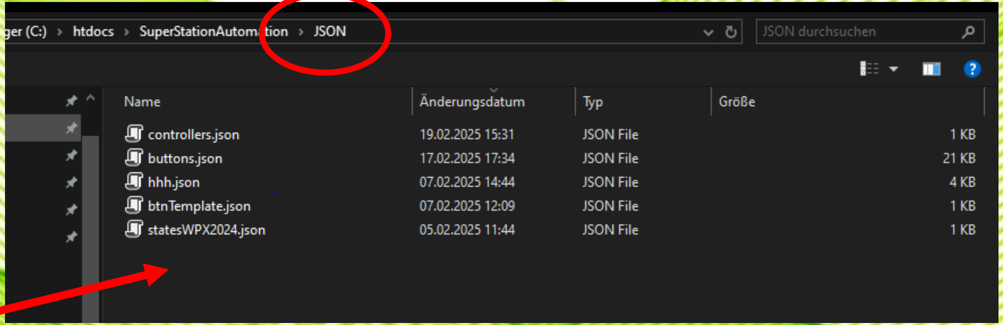
Edit a button config



*Load from local disc. Server see later



```
[ 32 items
  0 : {
    cmdSet : [ 1 item
      0 : {
        Cmd : s/0/8/1
        delayNext : 100
      }
    ]
    Mode : ws
    ip : 192.168.0.143
    title : PowerFlex ON
    alt : PowerFlex ON
    color : ☐ #222222
  }
  1 : {
    cmdSet : [ 1 item
      0 : {
        Cmd : s/0/9/1
        delayNext : 100
      }
    ]
    Mode : ws
    ip : 192.168.0.143
    title : PowerICOM ON
    alt : PowerICOM ON
    color : ☐ #222222
  }
  2 : {
    cmdSet : [ 1 item
      0 : {
        Cmd : s/0/10/1
        delayNext : 100
      }
    ]
    Mode : ws
```



Name	Änderungsdatum	Typ	Größe
controllers.json	19.02.2025 15:31	JSON File	1 KB
buttons.json	17.02.2025 17:34	JSON File	21 KB
hhh.json	07.02.2025 14:44	JSON File	4 KB
btnTemplate.json	07.02.2025 12:09	JSON File	1 KB
statesWPX2024.json	05.02.2025 11:44	JSON File	1 KB

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!

Super.Station.Automation

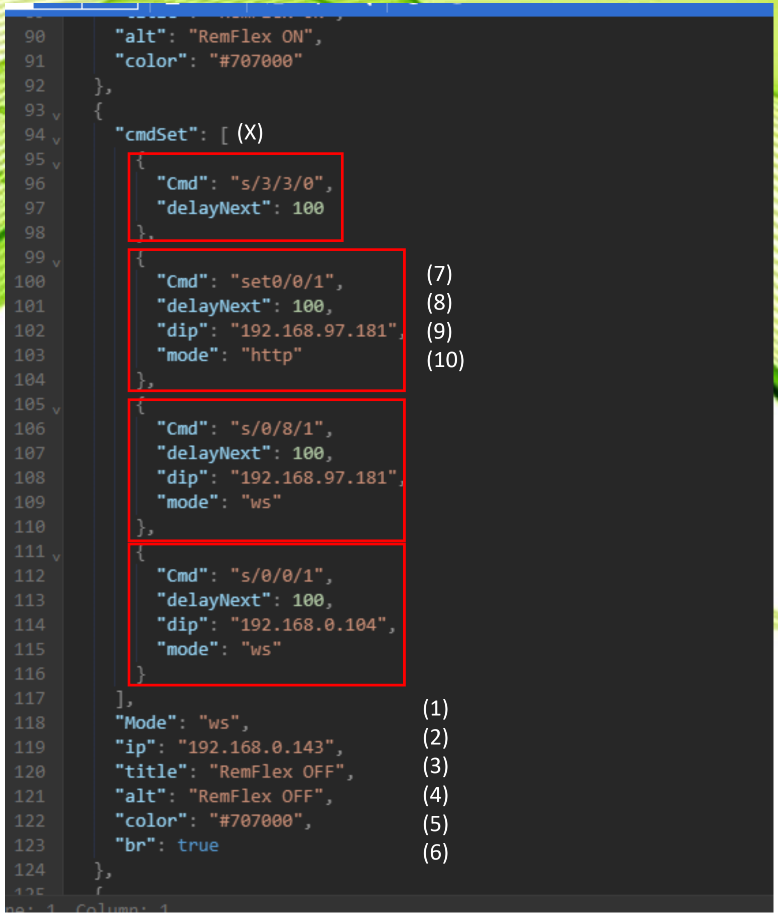
XX

Button Panel

1

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 ist common definition. The commands are chained in command set (X) with a Cmd (7), a delayed exec.in ms (8) and a differnt 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.



```
      "alt": "RemFlex ON",
      "color": "#707000"
    },
    {
      "cmdSet": [ (X)
        {
          "Cmd": "s/3/3/0",
          "delayNext": 100
        },
        {
          "Cmd": "set0/0/1",               (7)
          "delayNext": 100,                (8)
          "dip": "192.168.97.181",         (9)
          "mode": "http"                   (10)
        },
        {
          "Cmd": "s/0/8/1",
          "delayNext": 100,
          "dip": "192.168.97.181",
          "mode": "ws"
        },
        {
          "Cmd": "s/0/0/1",
          "delayNext": 100,
          "dip": "192.168.0.104",
          "mode": "ws"
        }
      ],
      "Mode": "ws",                        (1)
      "ip": "192.168.0.143",               (2)
      "title": "RemFlex OFF",              (3)
      "alt": "RemFlex OFF",                (4)
      "color": "#707000",                  (5)
      "br": true                           (6)
    },
    {
```

Hint: Make yourself familar 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

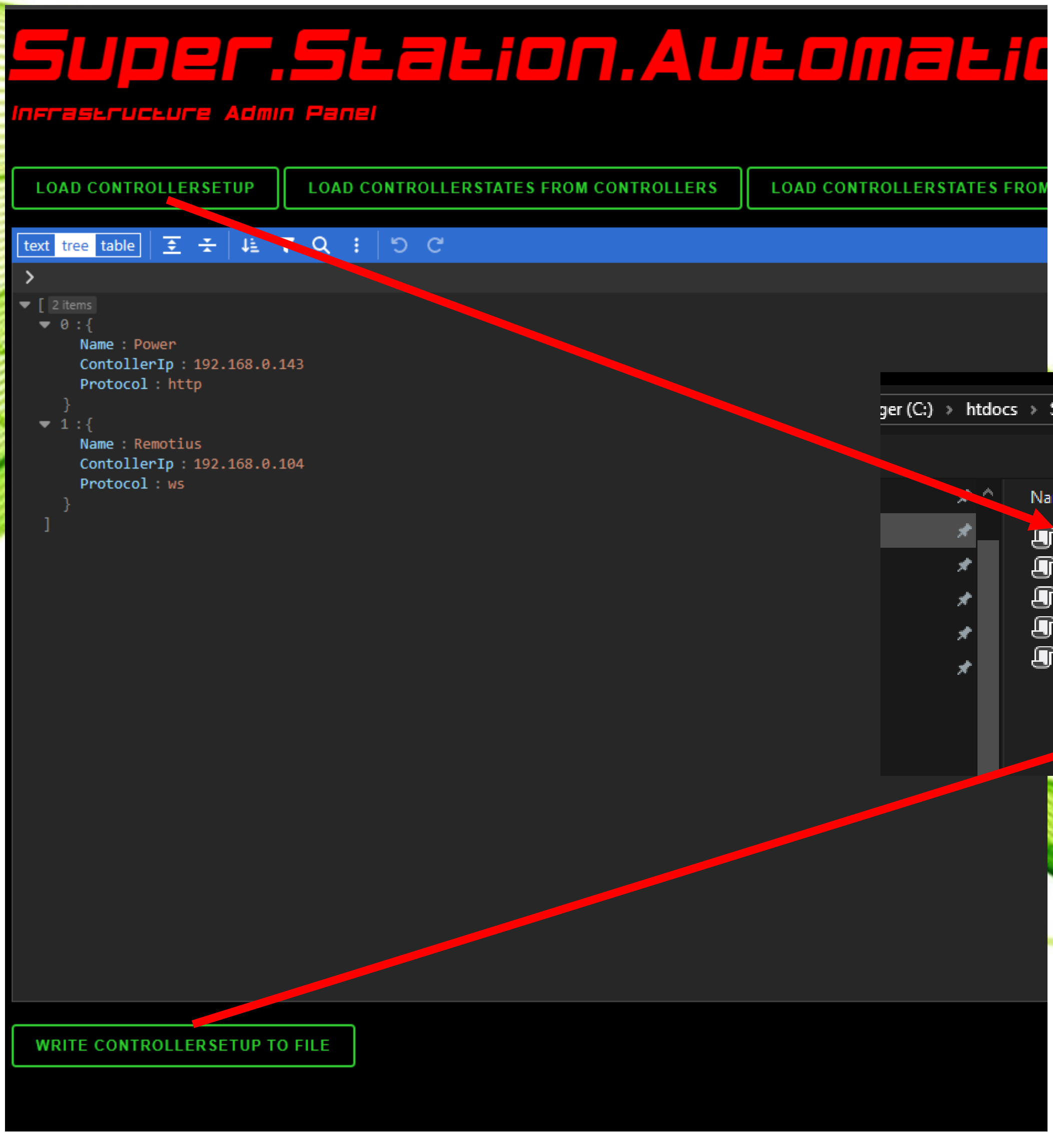
Super.Station.Automation

XX

Infrastructure Admin Panel

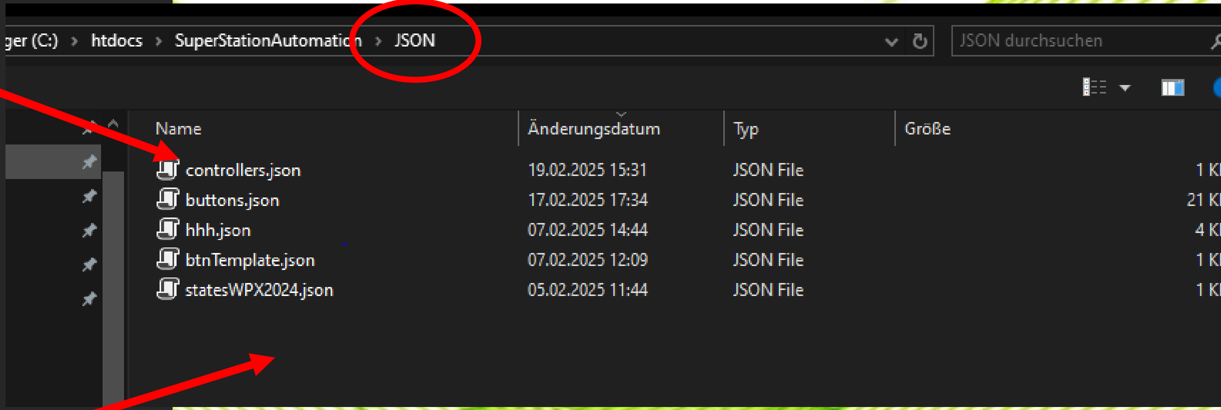
2

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



Super.Station.Automatic
Infrastructure Admin Panel
LOAD CONTROLLERSETUP
LOAD CONTROLLERSTATES FROM CONTROLLERS
LOAD CONTROLLERSTATES FROM
text
tree
table
WRITE CONTROLLERSETUP TO FILE

```
[ 2 items
  0 : {
      Name : Power
      ContollerIp : 192.168.0.143
      Protocol : http
  }
  1 : {
      Name : Remotius
      ContollerIp : 192.168.0.104
      Protocol : ws
  }
]
```



Name	Änderungsdatum	Typ	Größe
controllers.json	19.02.2025 15:31	JSON File	1 K
buttons.json	17.02.2025 17:34	JSON File	21 K
hhh.json	07.02.2025 14:44	JSON File	4 K
btnTemplate.json	07.02.2025 12:09	JSON File	1 K
statesWPX2024.json	05.02.2025 11:44	JSON File	1 K

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!

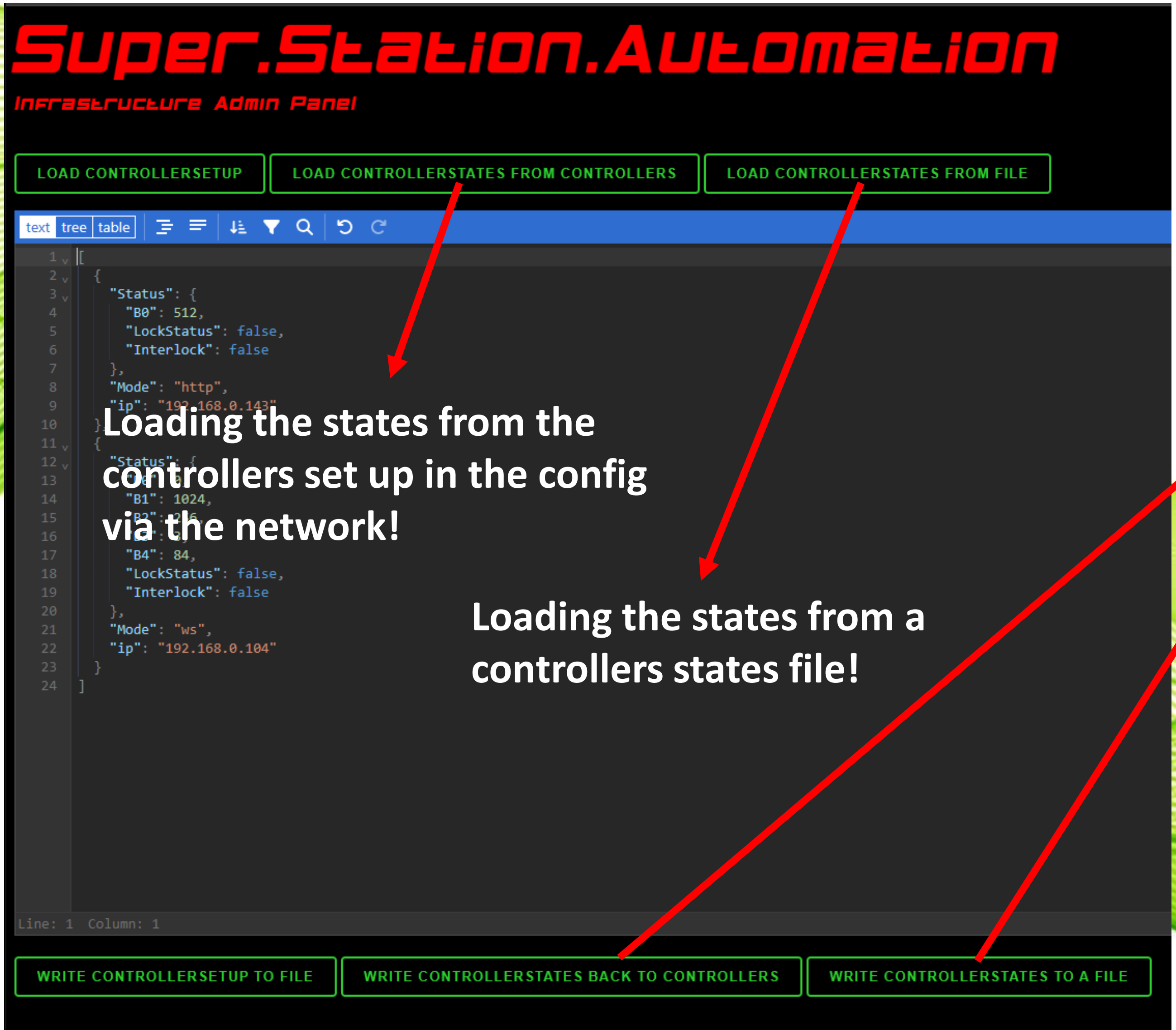
Super.Station.Automation

XX

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
Line: 1 Column: 1
WRITE CONTROLLERSETUP TO FILE
WRITE CONTROLLERSTATES BACK TO CONTROLLERS
WRITE CONTROLLERSTATES TO A FILE

```
[
  {
    "Status": {
      "B0": 512,
      "LockStatus": false,
      "Interlock": false
    },
    "Mode": "http",
    "ip": "192.168.0.143"
  },
  {
    "Status": {
      "B0": 0,
      "B1": 1024,
      "B2": 256,
      "B3": 5,
      "B4": 84,
      "LockStatus": false,
      "Interlock": false
    },
    "Mode": "ws",
    "ip": "192.168.0.104"
  }
]
```

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.

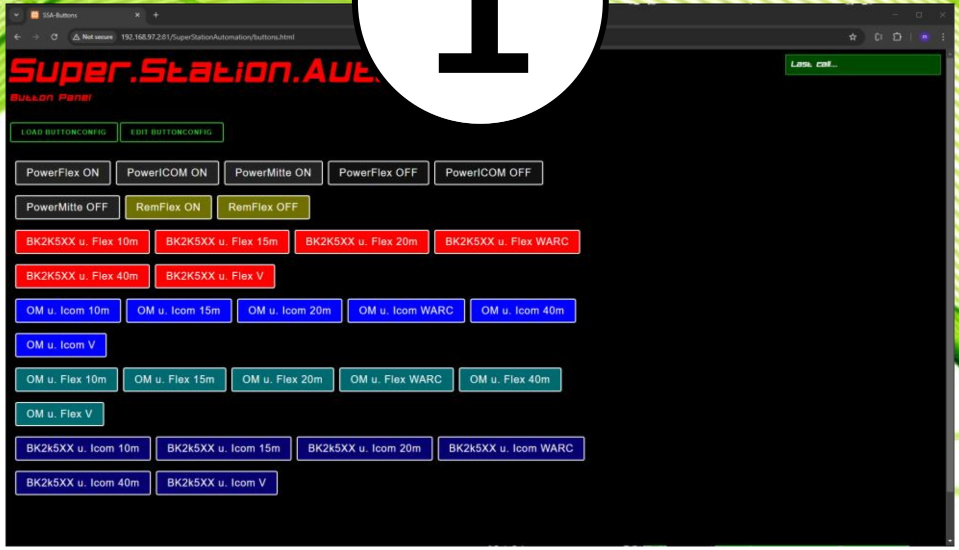
Super.Station.Automation

XX

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

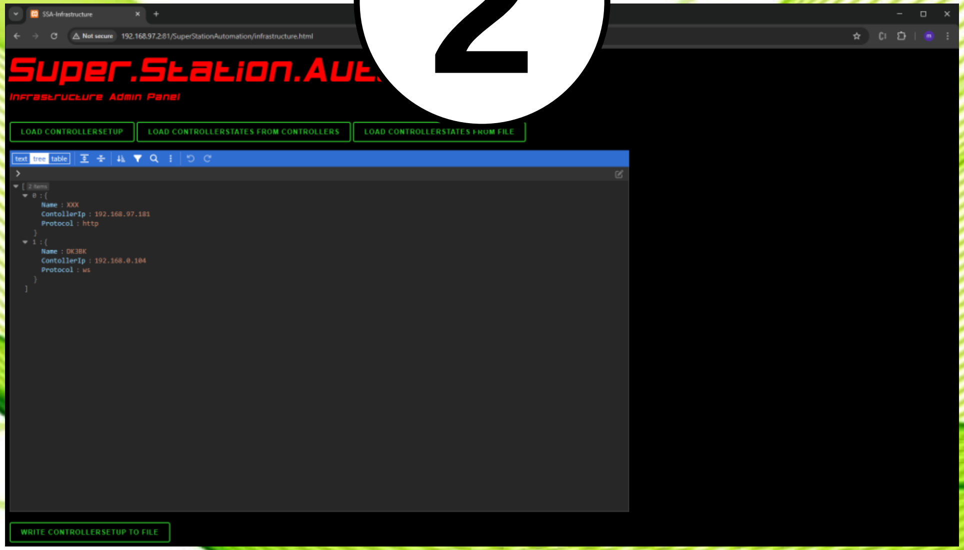
URLs/Pagenames for local use:

1



buttons.html

2



SSA-Infrastructure
Not secure 192.168.97.2:81/SuperStationAutomation/infrastructure.html
Super.Station.Aut
Infrastructure Admin Panel
LOAD CONTROLLERSETUP
LOAD CONTROLLERSTATES FROM CONTROLLERS
LOAD CONTROLLERSTATES FROM FILE
text
tree
table
WRITE CONTROLLERSETUP TO FILE

```
[ 2 items
  0 : {
    Name : XXX
    ContollerIp : 192.168.97.181
    Protocol : http
  }
  1 : {
    Name : DK3BK
    ContollerIp : 192.168.0.104
    Protocol : ws
  }
]
```

infrastructure.html

Super.Station.Automation

XX

Button Panel

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!

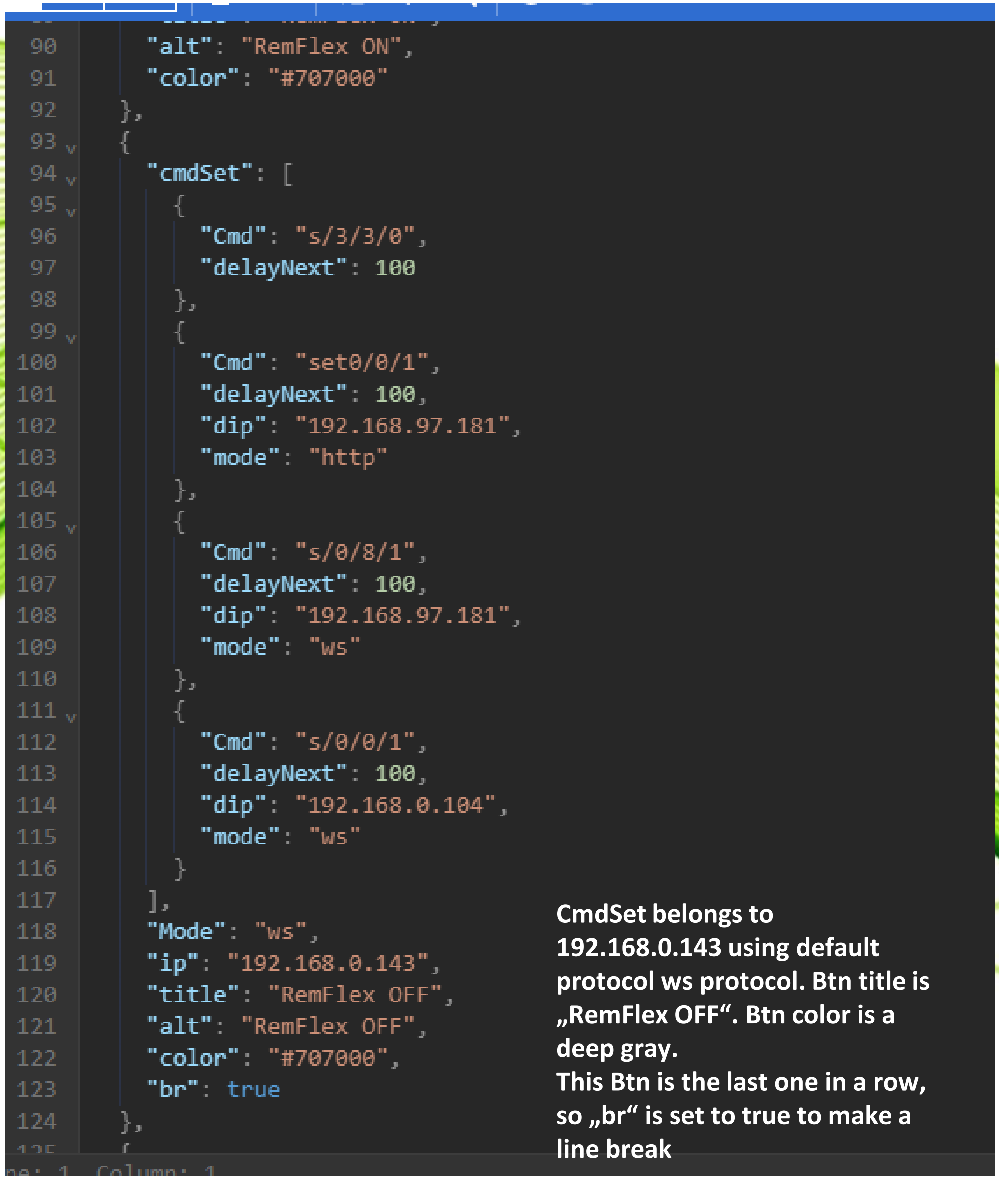
2

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

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

Super.Station.Automation XX

What does the example cmdSet do?



```
    "alt": "RemFlex ON",
    "color": "#707000"
  },
  {
    "cmdSet": [
      {
        "Cmd": "s/3/3/0",
        "delayNext": 100
      },
      {
        "Cmd": "set0/0/1",
        "delayNext": 100,
        "dip": "192.168.97.181",
        "mode": "http"
      },
      {
        "Cmd": "s/0/8/1",
        "delayNext": 100,
        "dip": "192.168.97.181",
        "mode": "ws"
      },
      {
        "Cmd": "s/0/0/1",
        "delayNext": 100,
        "dip": "192.168.0.104",
        "mode": "ws"
      }
    ],
    "Mode": "ws",
    "ip": "192.168.0.143",
    "title": "RemFlex OFF",
    "alt": "RemFlex OFF",
    "color": "#707000",
    "br": true
  },
```

CmdSet belongs to 192.168.0.143 using default protocol 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

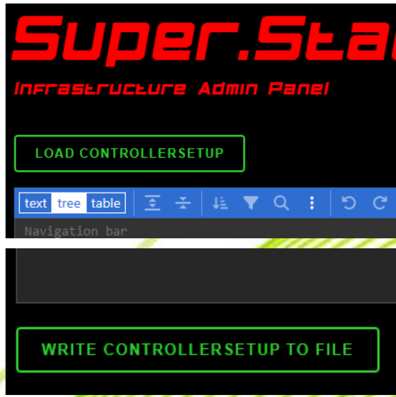
Need help? Ask for our setup service!

Super.Station.Automation

XX

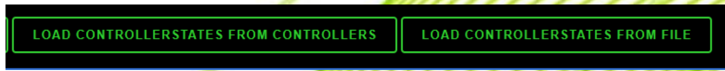
How to SAVE/RECALL controllerstates?

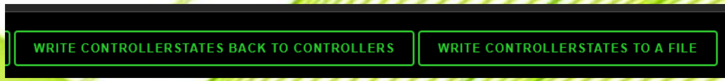
1. Make sure, all controllers who shd be part of your current setup (to be stored and recalled) are ONLINE/AVAIBLE/ON
2. Load your controllersetup
3. Edit the controllersetup, change settings, add or remove controllers
4. Save the Controllersetup to a file (overwrite or save as a different filename)
5. Reoopen the Infrastrucrue page (IMPORTANT!)
6. Load (a or the newly saved) controllersetup



SAVE the current controllerstates to a file:

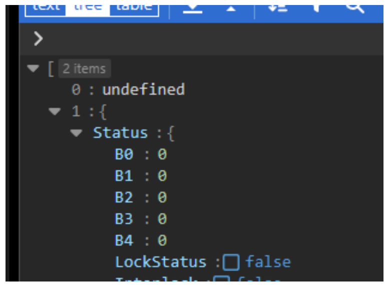
1. SAVE Workflow: Load the controllerstates from Controllers (IMPORTANT: ALL
2. SAVE Workflow: Write Controllersstates to a file


LOAD CONTROLLERSTATES FROM CONTROLLERS
LOAD CONTROLLERSTATES FROM FILE


WRITE CONTROLLERSTATES BACK TO CONTROLLERS
WRITE CONTROLLERSTATES TO A FILE

RECALL the current controllerstates to a file:

1. RECALL Workflow: Load Controllerstates from a file
2. RECALL Workflow: Write Controllerstates back to the controllers



```
[ 2 items
  0 : undefined
  1 : {
    Status : {
      B0 : 0
      B1 : 0
      B2 : 0
      B3 : 0
      B4 : 0
      LockStatus : false
```

Possible problems are all related that to the only reason: ONE or more OF YOUR CONTROLLERS IS/are OFFLINE!!!

You will identify this problem by:

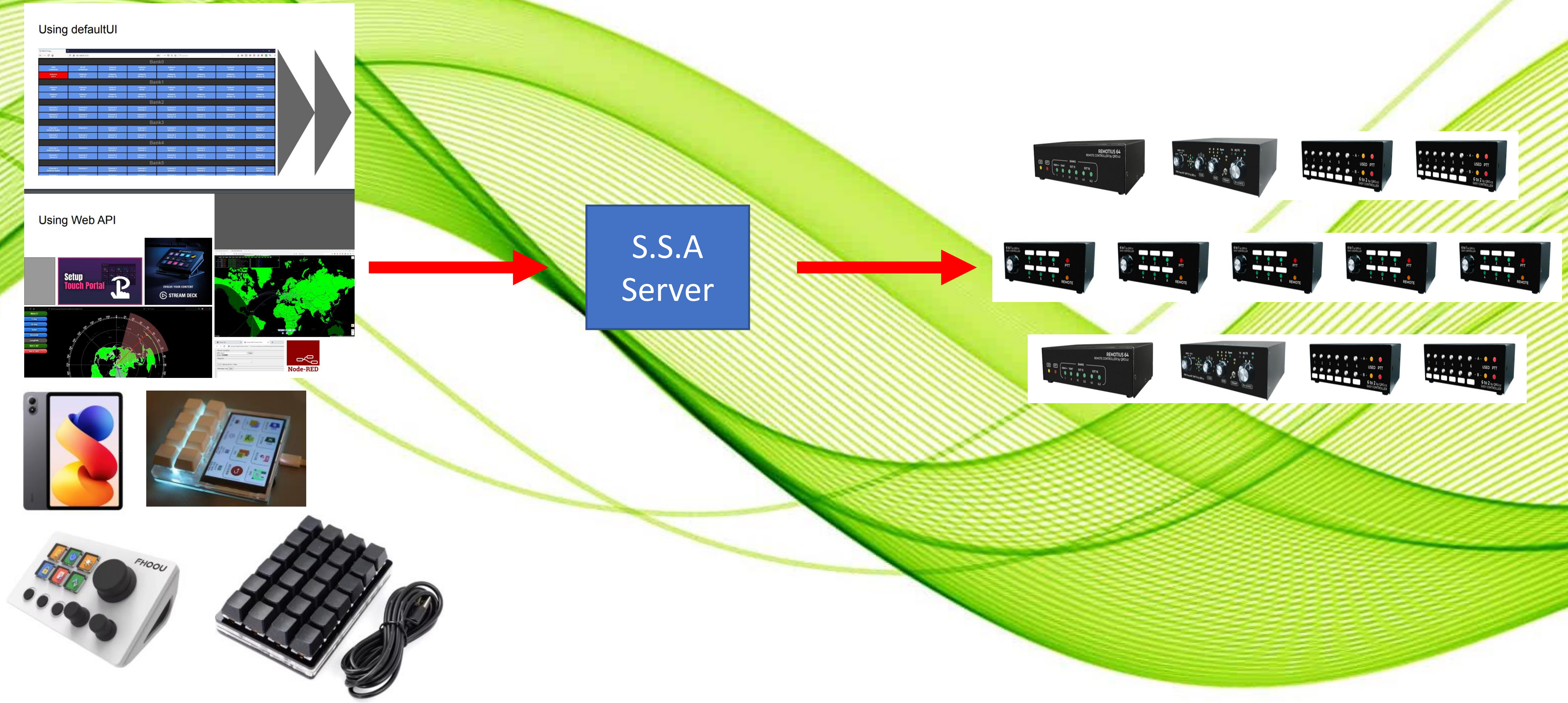
- click to „Load controllerstates…“ takes long to display something in the editor
- One or more controllers display „undefined“ as their status!

Super.Station.Automation XX

New: Super.Station.Automation *SERVER*

„Any Client, who can fire http-calls“

„Any of our qro.cz controller“



How does it work?

Example:
You a programmable keyboard to fire a cmd by button press.
Lets assume the cmd name is called „Go20FlexNA“.
Meaning: Flex_5, go to 20m and switch to NA-Antenna-Stackmatch and use the Stack 3x5elm.

The Super.Station.Automation-Server is listing for commands. He gets the command „Go20FlexNA“.
S.S.A-Server is searching in his lookup-list for a command-chain called „Go20FlexNA“.

He finds the command-chain and starts executing it (example):
Step1: Switch 2x6-Radio selector to use Flex_5
Step2: Switch ptt-manager to connect PTT with Flex_5 and RF-Power
Setp3: Connect OUTPUT of PA with 2x10 Antenna Selector, select NA-Stackmatch-Controller
Step4: Switch the Stackmatch controller to stack all 3 5elm-Yagis

All with just 1 click!
Execute as many commands as a chain from any http-device!
As many chains as you want!

Super.Station.Automation

XX

Which client hardware is supported?

Any client which can be programmed to fire http/https-commands like:

Your browser
Your tablet
Your smartphone
Your streamdeck
Your additional keyboard-keys
Your programmable keyboard
Your programable keys and rotary knobs controller

also your logging-software like N1MM or DXLOG!
…even your multimedia TV or Fridge!!! :P

Think about Super.Station.Automation! Only the sky is the limit!

Super.Station.Automation

XX

Special setup service!

We offer a speciel setup and confguration 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