

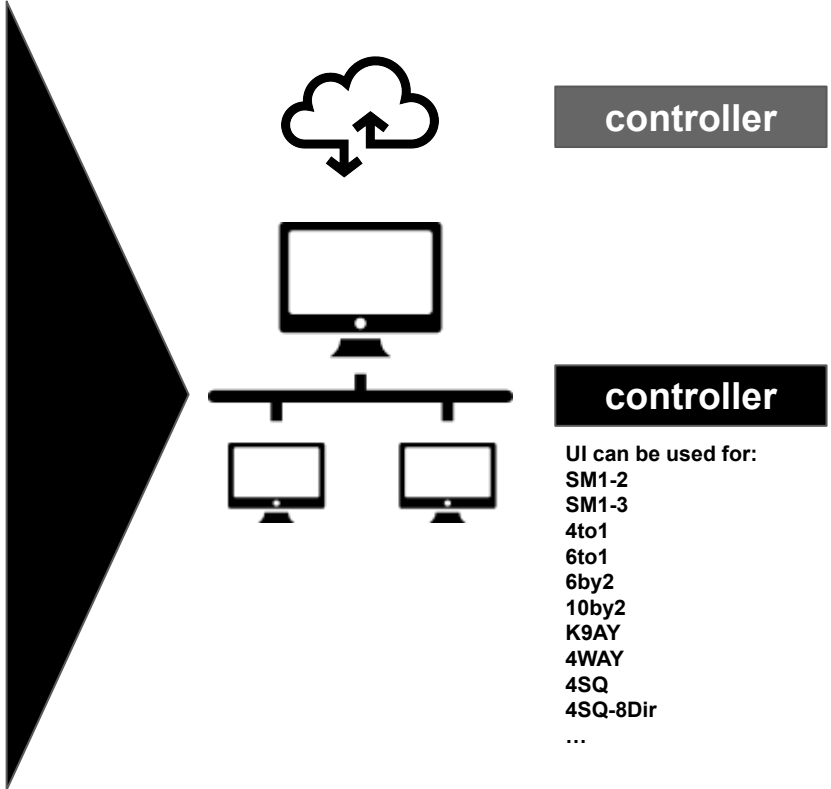
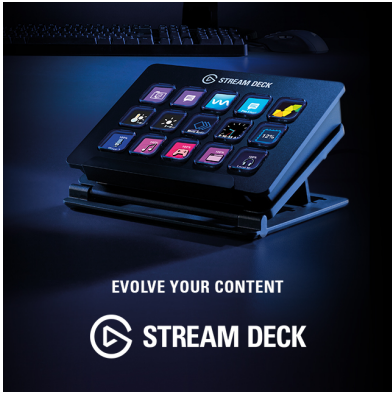
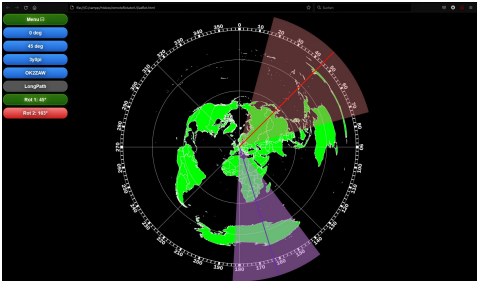
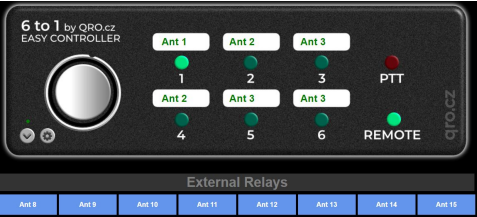
remoteSwitchFramework

most flexible web based architecture approach

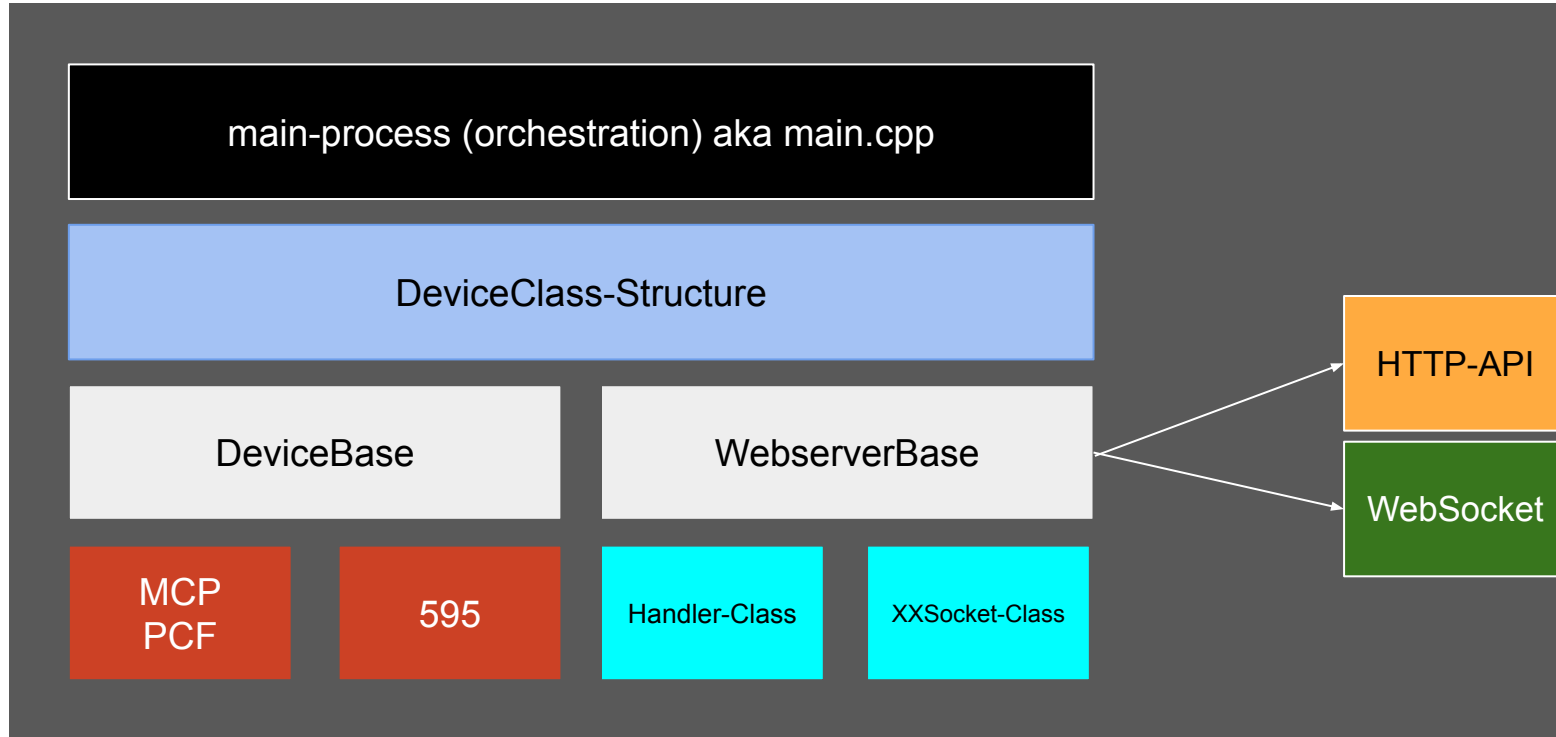
Flexibility counts

- don't overload the mcu with running a full webserver + serving all webfiles to the client. A mcu is a mcu and not a webserver. Don't interrupt the process of switching caused by webserver tasks delays.
- don't store any kind of businesslogic besides the process of switching on the mcu.
- follow important software architecture and design principles: separate hardware from process logic, separate process logic from presentation/ui
- keep the business logic under our control, so there is no need to upload the firmware of the mcu due to changes in the (webstored) business logic. Easy bug fixing, easy updates!
- provide a default set for serving the web files from qro.cz, but be most flexible where to re-route the files to any location needed.
- provide a default set of web pages for download/zip to be modified and stored to a (local) webserver
- new: provide a desktop app to spawn a webserver in the background. A user does not need to install a webserver
- Caveat: Solve the problems due to browser compatibility and security. Since this system is complex and support intensive, all files and firmware are copyright protected. This is not open source, but you as a customer of qro.cz will have all benefits using it.
- All qro.cz/remoteSwitch devices implement the same API for http and websockets. So it's easy to implement it for custom User interfaces, node red, Touch Portal or stream deck, your own Contest-Dashboard or mobile apps or or or...

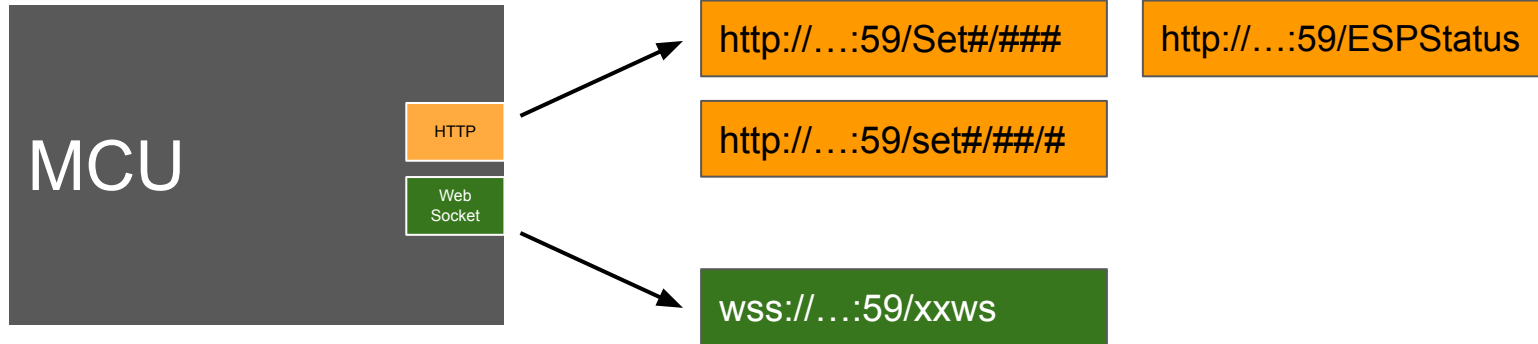
Decouple UI from hardware = most flexible, always up to date



Architecture in C++



Web API: http&ws



Simple Web API commands

http://<ip>:59...

/Get/
/Set<banknr>/<bitvalue>
/set<banknr>/<pinnr>/<0/1>
/Lock
/UnLock
/Reset
/ESPStatus

ws://<ip>:59/xxws

G
S/<banknr>/<bitvalue>
s/<banknr>/<pinnr>/<0/1>
L/<0/1>
R

Special implementation 6by2: S/s is X/x
X/<banknr>/<bitvalue>/<radionr>
s/<banknr>/<pinnr>/<0/1>/<radionr>

***Status requests and updates can be done by using http and/or websocket api.
Using the defaultUI, as configured in your custom config (json) file.***

Using defaultUI

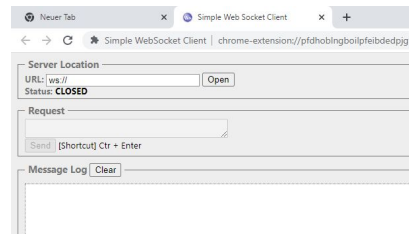
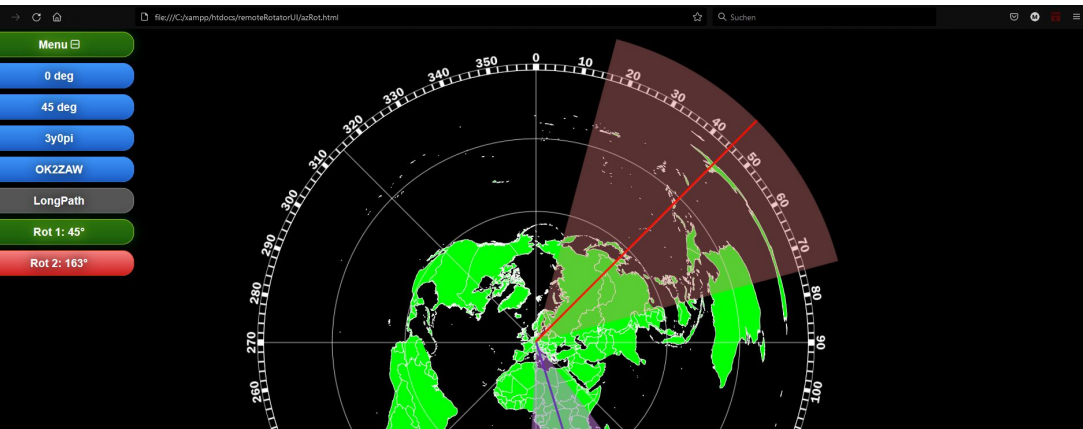
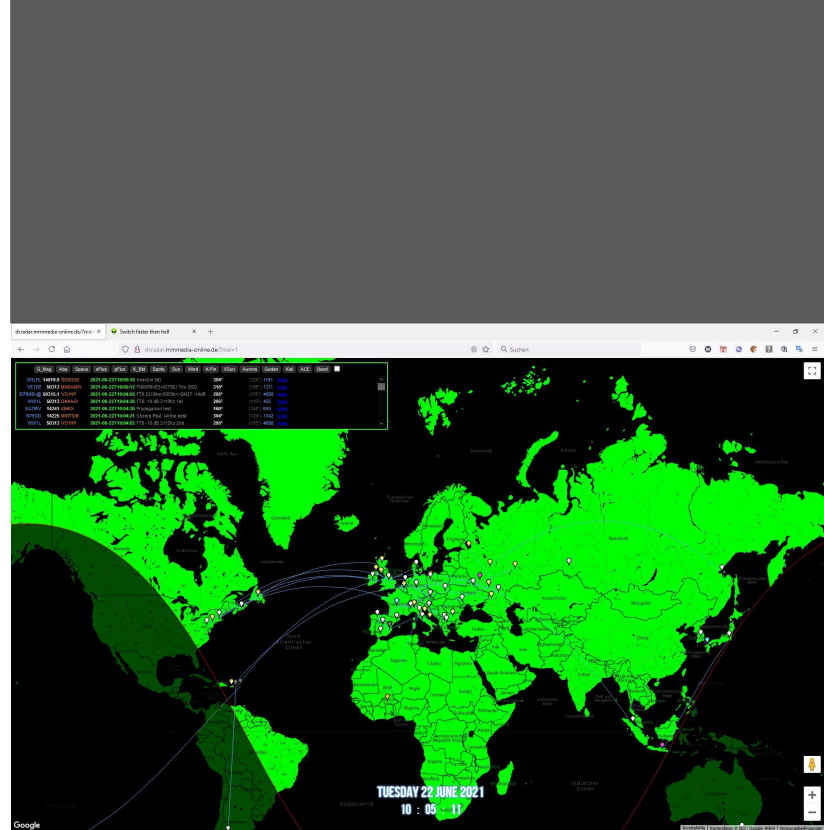
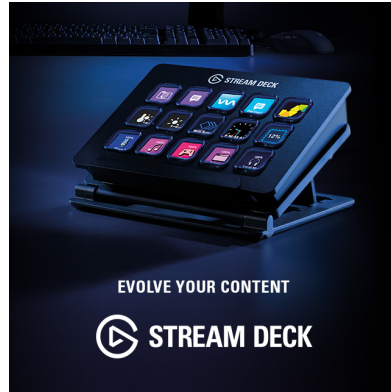
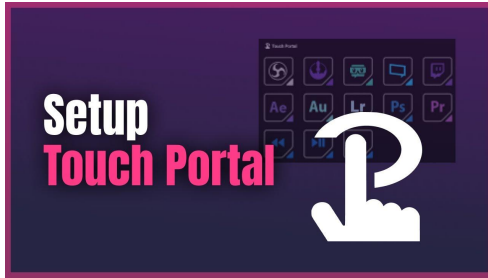


The screenshot shows a web browser window displaying a configuration page for a system with six banks (Bank0 to Bank5). The page is organized into a grid of blue cells, each containing text labels. The first cell in the 'Bank0' section is highlighted in red.

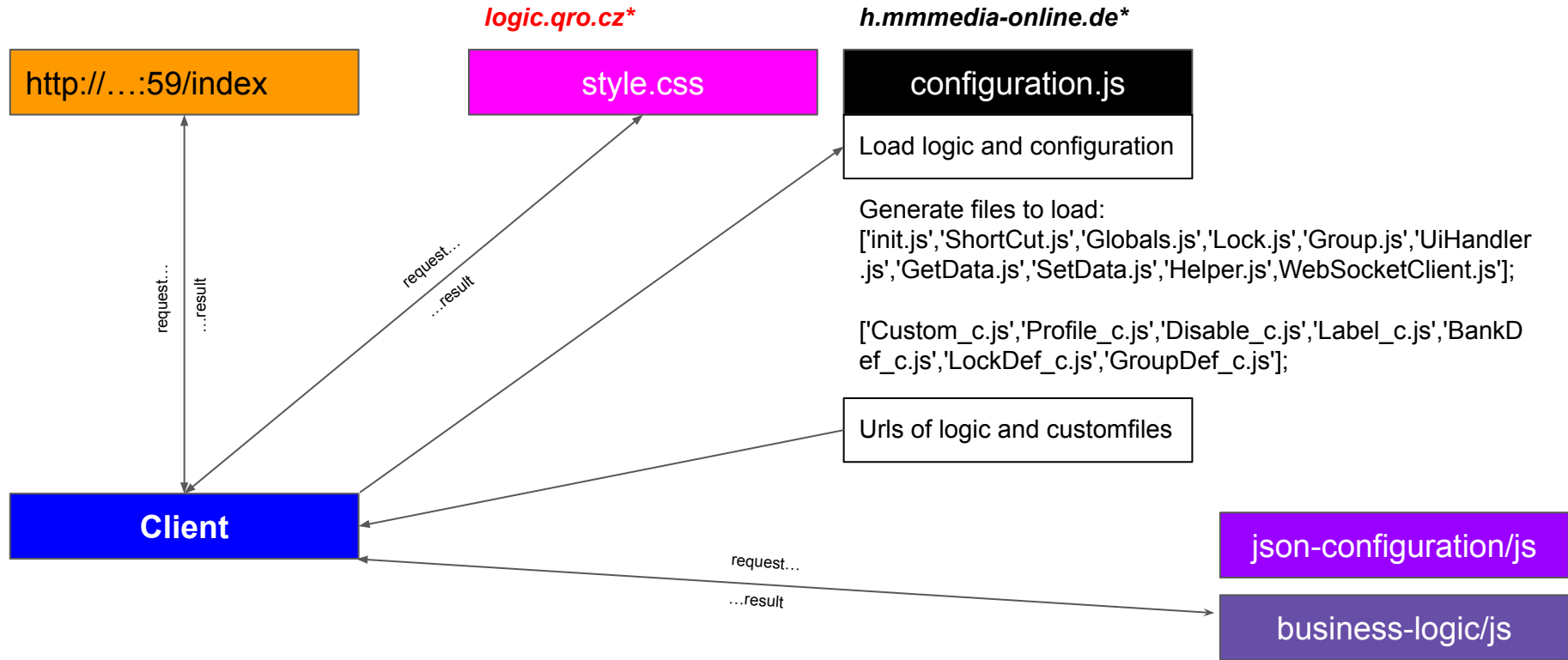
Bank0							
160m undefined	80 GP undefined	Antenna 80/40 D	Antenna 40 GP	Antenna Ant5	Antenna 20m	Antenna 17/15m	Antenna 12/10m
Antenna Ant 9	Antenna Ant 10	Antenna Device 10	Antenna Device 11	Antenna Device 12	Antenna Device 13	Antenna Device 14	Antenna Device 15
Bank1							
Antenna 160m	Antenna 80 GP	Antenna 80/40 D	Antenna 40 GP	Antenna Ant5	Antenna 20m	Antenna 17/15m	Antenna 12/10m
Antenna Ant 9	Antenna Ant 10	Antenna Device 10	Antenna Device 11	Antenna Device 12	Antenna Device 13	Antenna Device 14	Antenna Device 15
Bank2							
Channel 2 Device 0	Channel 2 Device 1	Channel 2 Device 2	Channel 2 Device 3	Channel 2 Device 4	Channel 2 Device 5	Channel 2 Device 6	Channel 2 Device 7
Channel 2 Device 8	Channel 2 Device 9	Channel 2 Device 10	Channel 2 Device 11	Channel 2 Device 12	Channel 2 Device 13	Channel 2 Device 14	Channel 2 Device 15
Bank3							
Channel 3 Antenne Super	Channel 3	Channel 3 Device 2	Channel 3 Device 3	Channel 3 Device 4	Channel 3 Device 5	Channel 3 Device 6	Channel 3 Device 7
Channel 3 Device 8	Channel 3 Device 9	Channel 3 Device 10	Channel 3 Device 11	Channel 3 Device 12	Channel 3 Device 13	Channel 3 Device 14	Channel 3 Device 15
Bank4							
Channel 3 Antenne Super	Channel 3	Channel 3 Device 2	Channel 3 Device 3	Channel 3 Device 4	Channel 3 Device 5	Channel 3 Device 6	Channel 3 Device 7
Channel 3 Device 8	Channel 3 Device 9	Channel 3 Device 10	Channel 3 Device 11	Channel 3 Device 12	Channel 3 Device 13	Channel 3 Device 14	Channel 3 Device 15
Bank5							
Channel 3 Antenne Super	Channel 3	Channel 3 Device 2	Channel 3 Device 3	Channel 3 Device 4	Channel 3 Device 5	Channel 3 Device 6	Channel 3 Device 7
Channel 3 Device 8	Channel 3 Device 9	Channel 3 Device 10	Channel 3 Device 11	Channel 3 Device 12	Channel 3 Device 13	Channel 3 Device 14	Channel 3 Device 15



Using Web API



Using defaultUI: ...:59/index

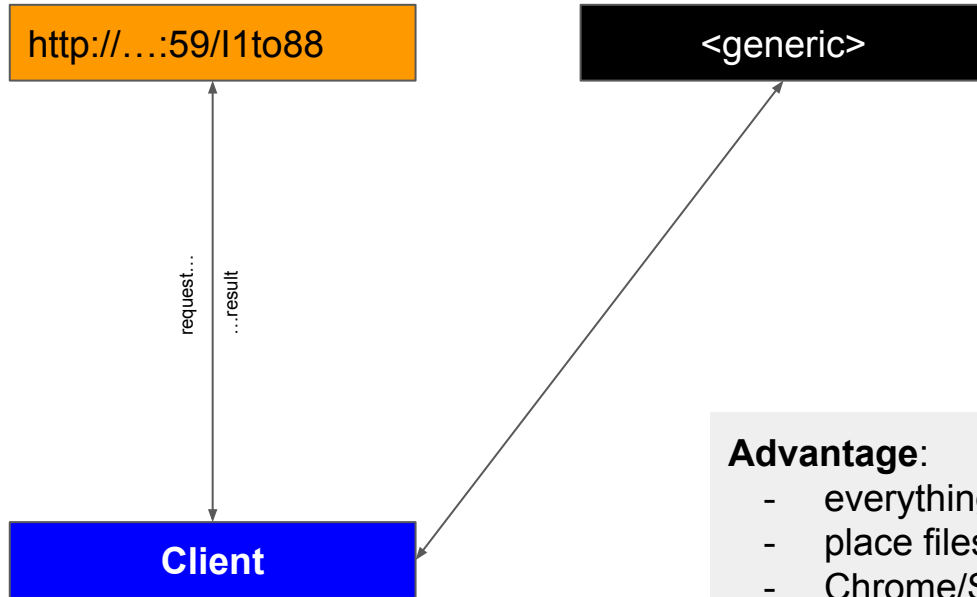


*default can be changed by paramaters**
*fixed path to configuration.js**

Using defaultUI: ...:59/index?<parameters>

t	<typeidentifier> for dominator (/l1to88)
ptd	url to full generic generator (/lto88)
css	url to css file (configuration.js)
redirect	redirect to specified url (configuration.js)
ipofswitch	(local) ip or url of the switch (configuration.js) (without protocol)
portofswitch	port 59 as default firmware port (configuration.js)
jsurl	url where all the js files are located (configuration.js) (without protocol)
customizeurl	url where all config json files are located (configuration.js) (without protocol)
nrofboards	number of 16ch banks (only caller.html, see “files on local computer”)

Using defaultUI: ...:59/l1to88



Purpose:

<generic>-Page (js or php, etc.) is responsible to inject the urls for the needed businesslogic and custom-json files.

Also the url for css is generated and additional functions called (like init() in onload())

Advantage:

- everything you want can be injected.
- place files wherever you want (local/sever)
- Chrome/Security problems solved since "switchip" serving the html page

Using defaultUI: ...:59/l1to88?<parameters>

t <typeidentifier> for generic generator like dominator.php (/l1to88)
ptd url to full generic generator (/l1to88) (incl. protocol)

+ all parameters of the generated urls to js-files

Using l1to88 solves the issue, that all pages should be served by the webserver, in this case the impero (Cave eat). Issues with access private network can be solved. And this is the most flexible way of page generation by javascript. All is dynamic :)

Restrictions and know issues

CORS - Cross origin requests

Affected: All browsers. Chrome vy restrictive.

Solution: Add header in firmware (done).

Use /I1to88, App, VPN, Firefox or Cors-Browser-Plugin if running into issues

APPLE IOS or OSX

Officially not supported since there is no OSX device available. DefaultUI might work since it was tested on IPad, but more complex special UI for qro-devices using Web API might not. Get in touch in case of issues.

Supported browser is Firefox

Since this is a real complex Solution, focus was on one browser with best platform compatibility. Chrome might work, too, but not for the pinger.html for example... Please report bugs to [remoteswitch\(at\)gmx.de](mailto:remoteswitch(at)gmx.de)

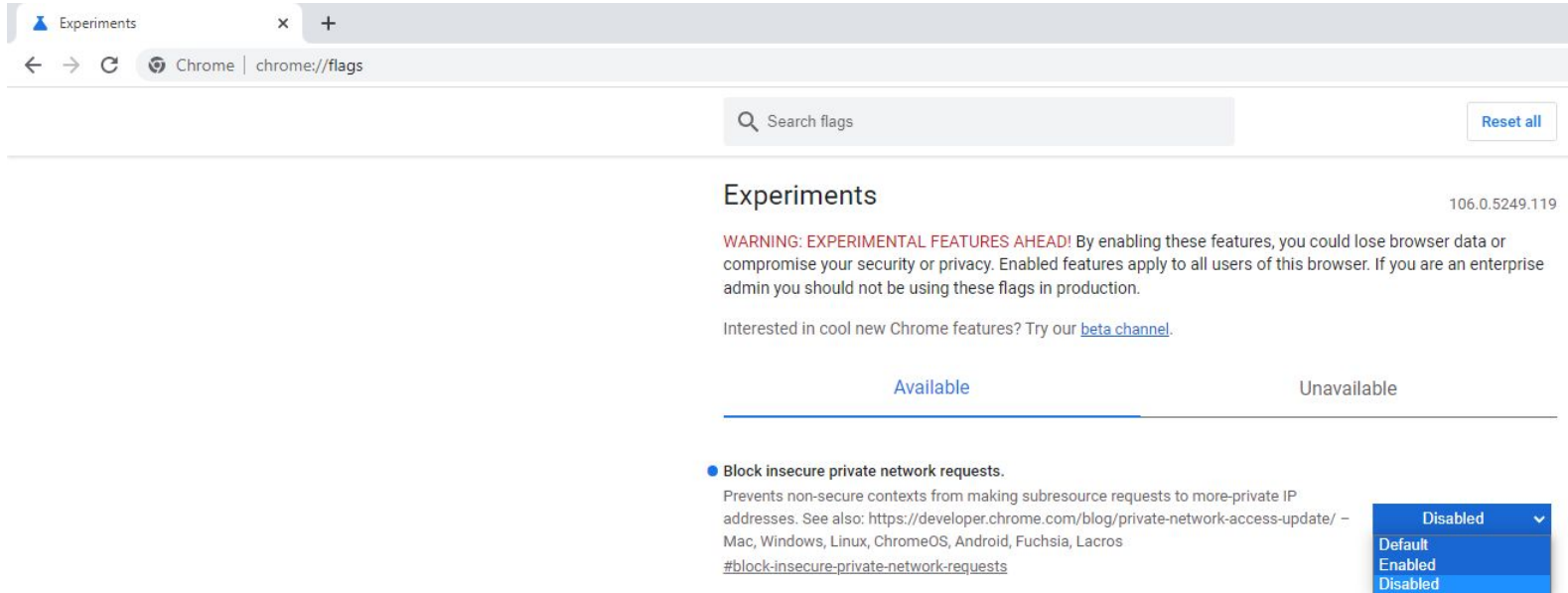
https

A small mcu is unfortunately in short of memory. The certificates and encryption takes a lot of performance. So using a VPN is really recommended in general anyway, all traffic is http. Win-win-win for all. There might be more an more issues with browser restriction, so using a https to http gateway might be useful one day. We keep our eye on that issue!

Restrictions and know issues

Upcoming: F***ing Chrome implementation of RFC preventing CSRF

Chrome decided to go his own way with private network access. Using the /I1to88 system will help here, but atm you still can do this if you are forced to use chrome (maybe older portable versions will help,too):



The screenshot shows the Chrome flags page at chrome://flags. The page title is 'Experiments' and the address bar shows 'chrome://flags'. A search bar is present with the text 'Search flags' and a 'Reset all' button. The main content area is titled 'Experiments' and shows a warning: 'WARNING: EXPERIMENTAL FEATURES AHEAD! By enabling these features, you could lose browser data or compromise your security or privacy. Enabled features apply to all users of this browser. If you are an enterprise admin you should not be using these flags in production.' Below the warning, there is a link to the 'beta channel'. The page is divided into two columns: 'Available' and 'Unavailable'. The 'Available' column contains a list of flags, with the first one being 'Block insecure private network requests'. This flag is currently set to 'Disabled'. A dropdown menu is open for this flag, showing the options 'Default', 'Enabled', and 'Disabled'. The 'Default' option is selected.

Experiments 106.0.5249.119

WARNING: EXPERIMENTAL FEATURES AHEAD! By enabling these features, you could lose browser data or compromise your security or privacy. Enabled features apply to all users of this browser. If you are an enterprise admin you should not be using these flags in production.

Interested in cool new Chrome features? Try our [beta channel](#).

Available Unavailable

- **Block insecure private network requests.**
Prevents non-secure contexts from making subresource requests to more-private IP addresses. See also: <https://developer.chrome.com/blog/private-network-access-update/> - Mac, Windows, Linux, ChromeOS, Android, Fuchsia, Lacros
[#block-insecure-private-network-requests](#)

Disabled
Default
Enabled
Disabled

Option (A) Using Files on your local Computer

Option (D) Using your own webserver (local/public)