ALTUI Plugin for VERA / Lite / Edge on UI5 or UI7

Micasaverde /GetVera is producing a product family of zWave controllers called Vera 3, Vera Lite and Vera Edge now. These product come with a user interface layer called UI and which exists in 2 versions as we speak : UI5 and UI7.

Unfortunately the long awaited UI7 has been kind of a disappointment , especially with its unresponsiveness , not really well optimized screen real estate, and difficult to deal with for 3rd party plug in writers. It was also promising a mobile user interface and the application does not resize well on phone or on desktops and the mobile version of the application is not user friendly.

I started to work on a UI replacement.

This document covers:

1. The overall project objectives & "big" rules
2. The screen shots
3. The installation instructions
4. Some internal explanation of the source code & architecture

Table of Contents

The project initial objectives

I am not fully satisfied with UI5 or UI7 and I think we can provide very quick improvement. French users of Orange HomeLive system on internet seems to be Highly largely unsatisfied by the UI and we could improve this relatively easily adopting a refreshed approach & architecture. (remains to be seen if orange is going to be open to this but we should try)

Objectives

1. **Fast & immediately responsive** (except LUA Jobs of course, cant control that). Asynchronous / threaded programming as much as possible.
2. Avoid the classic UIx issues with too many **heterogeneous js frameworks**, inconsistent CSS rules requiring ticks all over the place and overuse of the "! Important"
3. Does **not require anything** other infrastructure than the **VERA** itself and a simple plugin. No PHP, No DB, No additional server running. Just plain VERA
4. Really use the **power of the client side** machine (big processors & memory) and far less the Vera side.
5. Works well on all screen size**, full responsive design** using bootstrap
6. Really use **bootstrap** facilities for responsiveness, should work on Phone 4S as well as tablets, as well as desktop / large screens. Same app, same code, same access url
7. Dashboards should be optimized in screen real estate. **Undo the UI5 design decision** which links the Scene editor with the dashboard. You can only put in scene what is in the dashboard (unless you use the advanced feature). Dashboard requires maximum use of the screen real estate, Scene editor is something else.
8. Plugin authors should be able to control the display of their device using a s**imple javascript functions** , even on the dashboard page. Should not be limited to a VERA Box api or any complex undocumented json behaviors. Just a dynamically loaded javascript which can make full use of bootstrap & useful libraries provided
9. **Full reuse of dynamic icons** (don t want to recreate icons or each plugin logics here) from the json descriptions
10. **Dashboards should customizable by the end user,** he choses the pages and the devices he wants to see (not done yet at this point !)
11. Later on , more features, reuse of UI7 json descriptions for dashboards, control buttons etc if possible. To be investigated
12. Works on UI5 and UI7 with minor degradation on UI5 (housemode for instance)

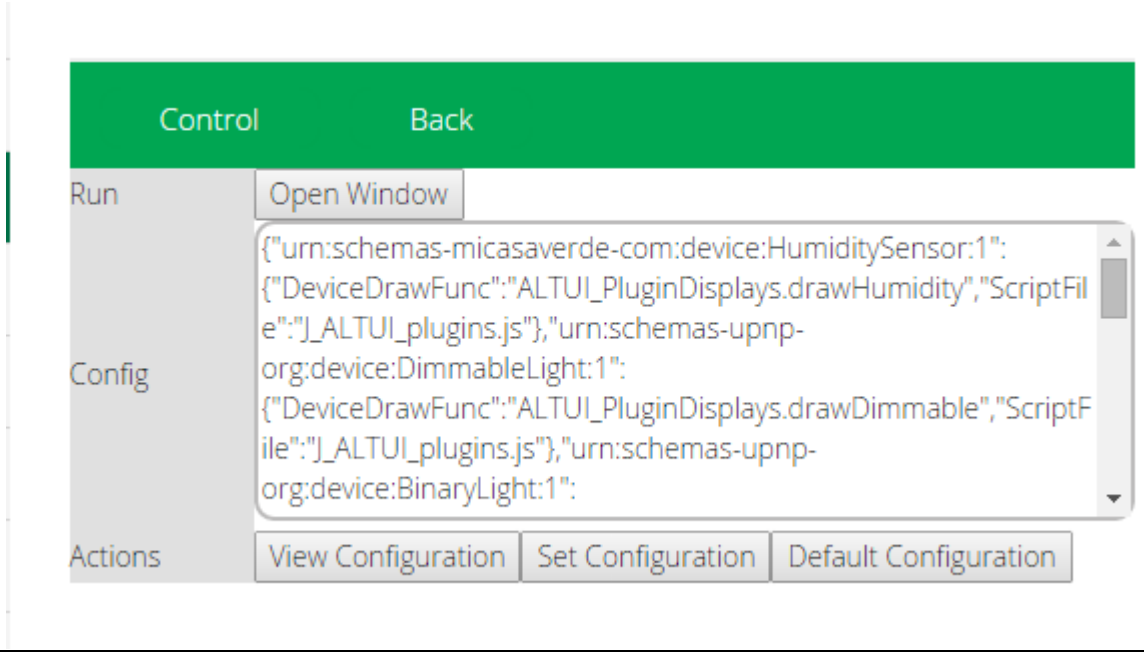
DONE and functional so far – VERSION 0.29

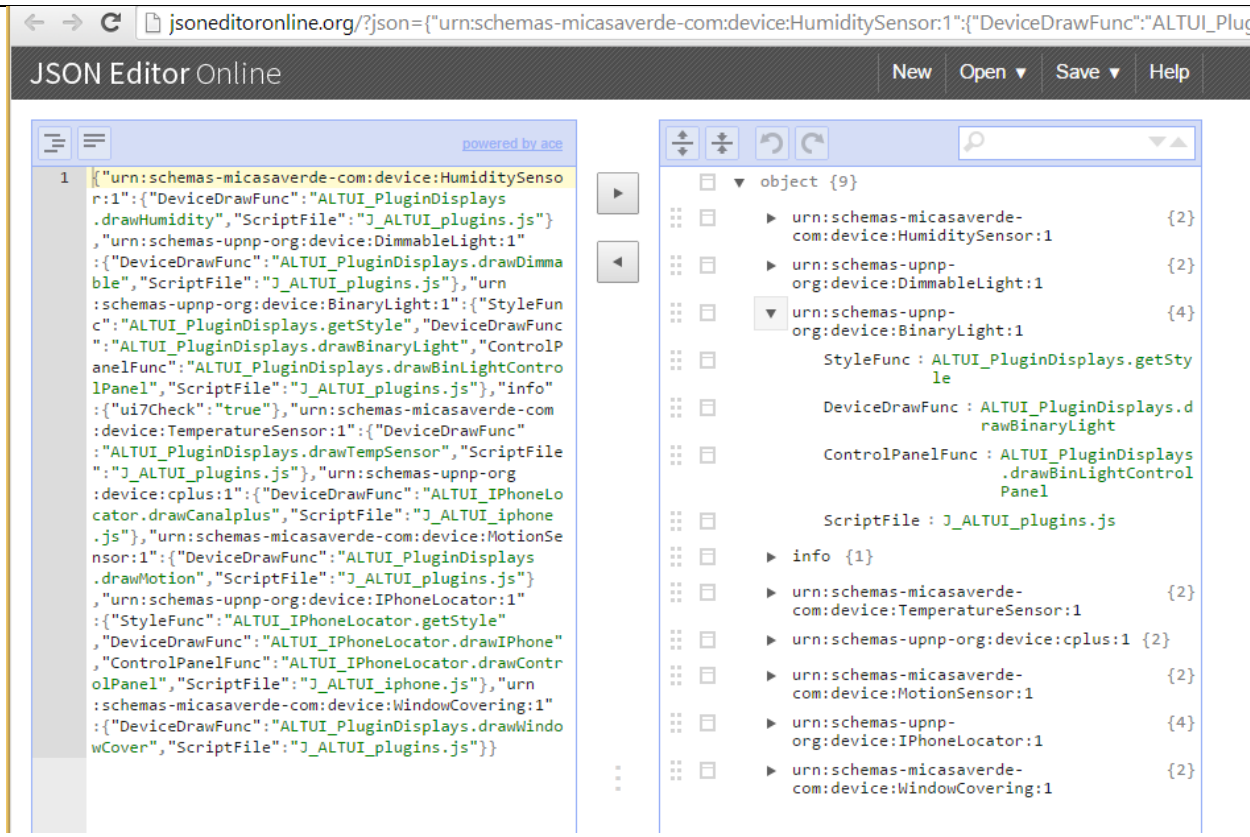
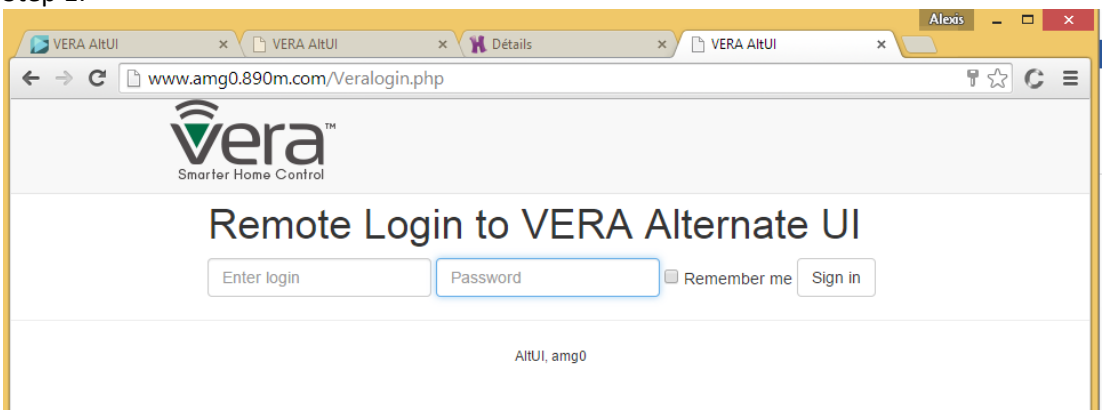
I already uses this more than UI7 on my ipad/phone and desktop at home. So far achievements are the following:

- Single plugin to install. Then access via the setting tab, or directly via : http://<veraip>/port_3480/data_request?id=lr_ALTUI_Handler&command=home#
- **Remote access login** via a PHP page located at http://amg0.site11.com/Veralogin.php or http://www.amg0.890m.com/Veralogin.php
    - Un like UI5 UI7, All plugins custom ICONS are working on remote access screen ! The backend servers of VERA (MMS) are not transporting/caching custom icons. This application uses a different technique, the plugin handler transmit necessary images as data URI and the client caches them
- Full responsive design, bootstrap & jquery based . check on your iPad when you rotate from portrait to landscape, it is quite fun , it just adds a column
- HouseMode view & change on ui7
- Footer display of VERA parameters (serial etc)
- List / Delete Rooms
- List devices with ultra simple and small dashboard
    - Device drawing can be customized by an individual js file. List of files to load is hard coded in the .LUA plugin code file but I ll change that later
    - Device Variable display : alphabetically sorted, htmlEncoded (so xml & json appear properly) , humanly readable timestamps.
    - Device variable Edit capability : click on it, modify, click out (it will save it automatically)
    - Device UPNP action dialog to dynamically retrieve actions & parameters names and enable the user to trigger action from this dialog.
    - Device Control Panel for a full page drawing customizable by device type dynamically. By default if no plugin is provided to customize the control panel , **the control panel page will display the same "control" tab as the pluging is doing on the VERA box** UI5 UI7 (the "flash" tab in the JSON device settings). It will try to emulate the vera device control placement rules as much as possible so the look & feel is similar to what the plugin author intended. If that is not enough or if a finer grained customization is wished in the user interface, the full ability to write a custom UI in a dynamically loaded javascript module is possible
    - Ability to filter devices on the device page by
        - per Room of devices
        - per Category of Device
        - per Battery device or not
        - per Visible /Invisible status
        - by name (filter "as you type" a name to search)
    - Display / color of Battery level
    - Room filtering selection in the left bar

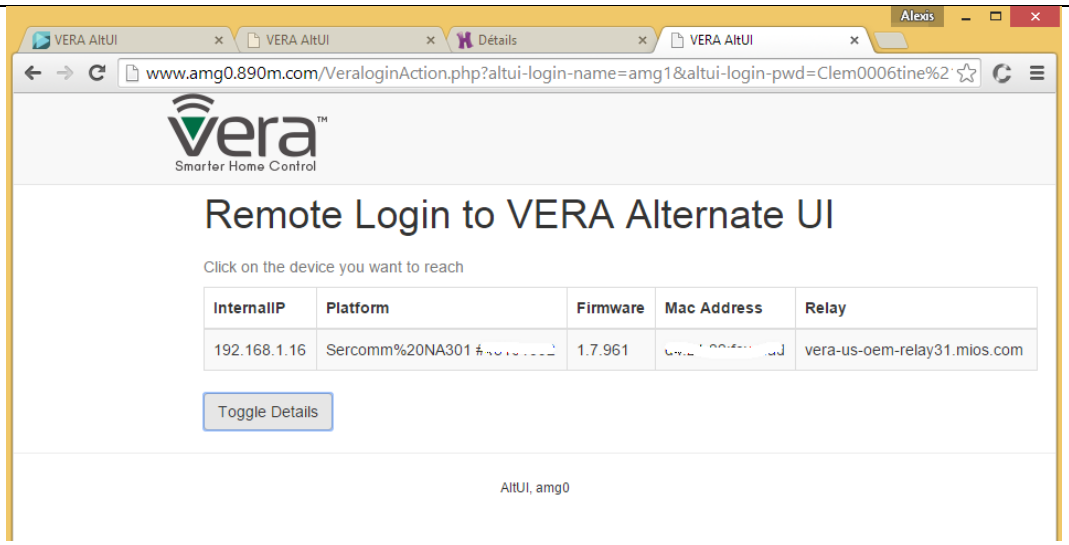
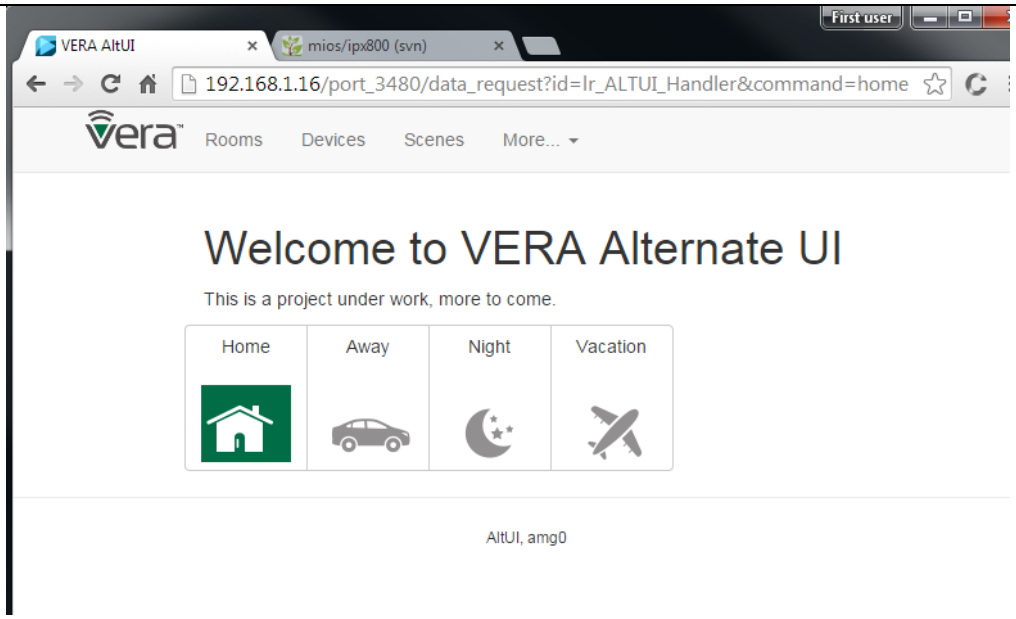
- List Delete , Edit, Run Scene
    - Full display of the scene parameters & lua code
    - Display of last run, next rune timestamp
    - Room filtering selection in the left bar
    - Edit features :
        - enable/disable & delete for triggers/timers/actions/groups and room assignment.
        - Delete individual triggers/timers/actions/groups
        - Add/edit Lua code for event triggers (UI7 does not have this capability, UI5 used to have it)
    - Wip…
- 2 fully asynchronous engines : user_data/l_data processing engine and UI refresh engine.
- See device dashboard and can filter by room
    - Implemented :
        - powerswitch, humidity, temperature, dimmers, door lock, door sensor, window covers
        - example of custom 3rd party plugin display: iphonelocator, cplus,
        - Motion sensor (arm / trip status)
        - all device icons are working (even if dynamic) including 3rd party plugin, including old UI5 mode ones
    - Not yet :
        - camera,
        - 3rd party plugin (but if authors are interested it is very easy, checkout the IphoneLocator .JS file)
- Plugins
    - Icon, name and version
    - List of installed 3rd party plugins with version and button to go directly to
        - Author help page
        - MIOS App Store page for the plugin
        - Single button to update to latest version
- Lua restart command in the menu
- Lua startup code edition
- Lua test code dialog box (broken feature on UI7 chrome)
- User control for the cache of icon & files in persistent storage (local storage HTML5 so persistent on a given machine). User can save or clear the cache. It will avoid redownloading icons unecessarly
- Credit page

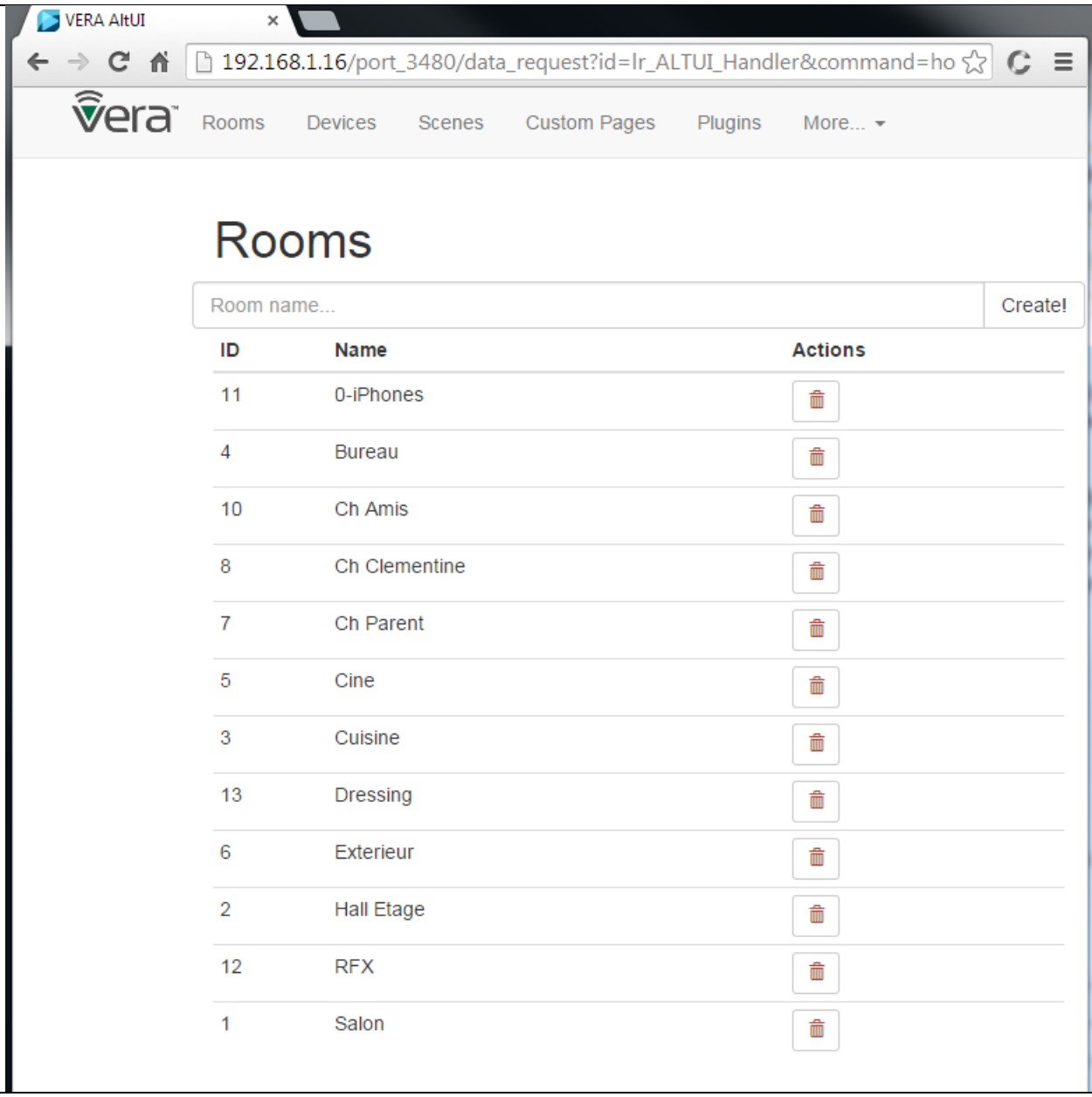
Screen shots:

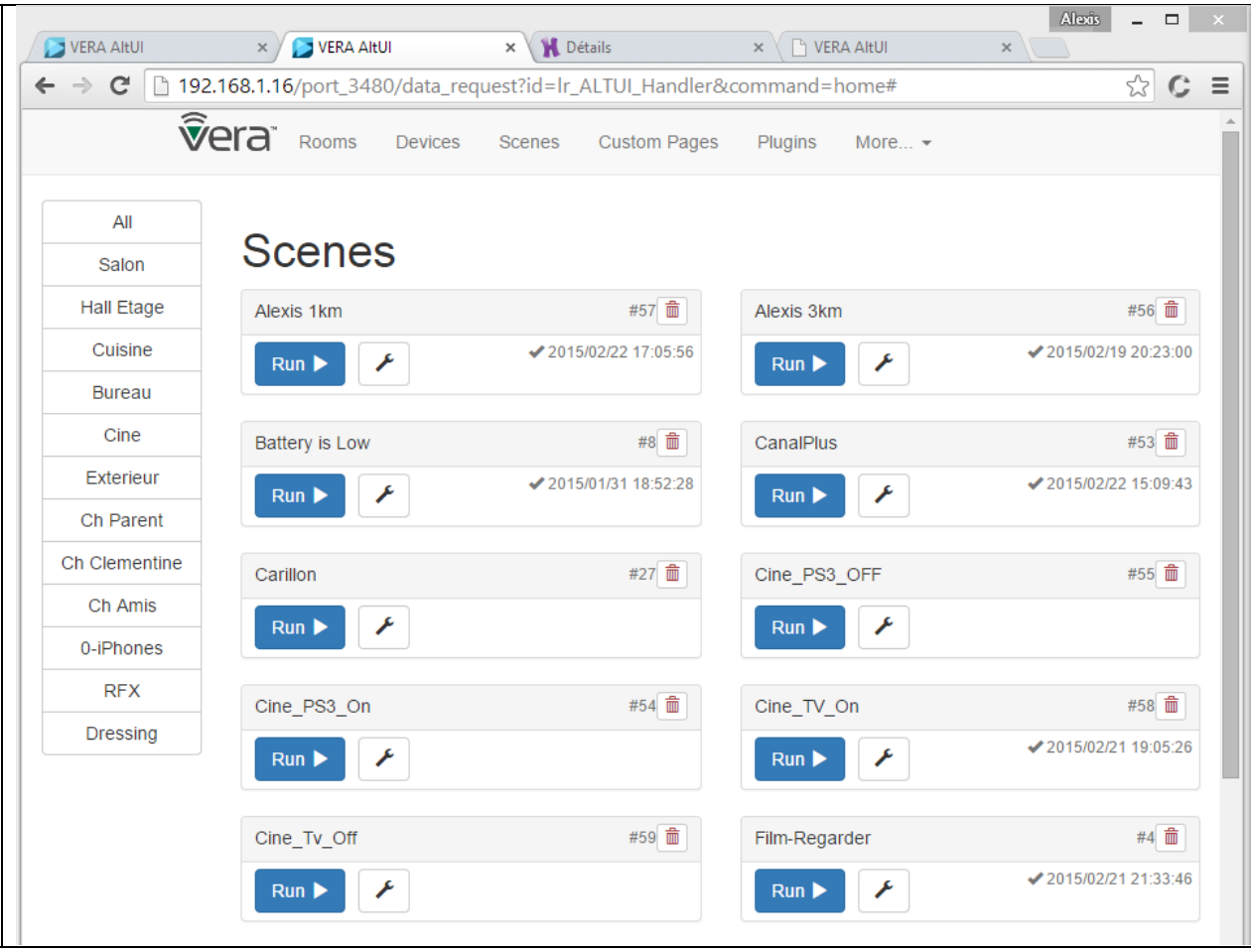
LUA plugin	Plugin Setting for configuration : Open => launch the ALTUI window Dynamic configuration for additional modules/plugin display functions Reset config to default Open & View configuration in a online json viewer 

<table>
<tr>
<td></td>
<td> 
</td>
</tr>
<tr>
<td>Remove Access via MMS servers for UI7 boxes</td>
<td>Step 1:
 

Step2:</td>
</tr>
</table>

	
Home mode selection	

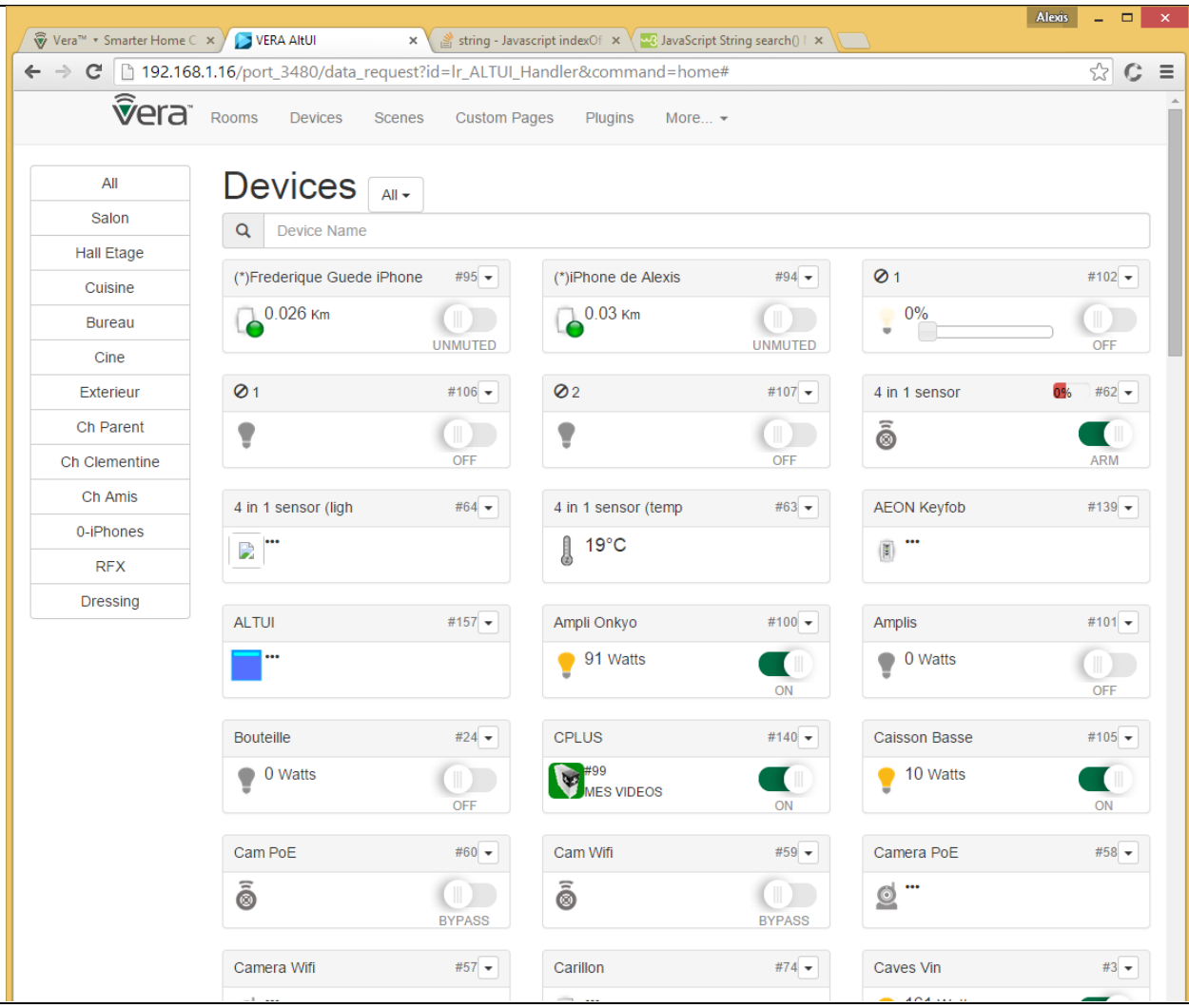
Room list and create/delete actions	

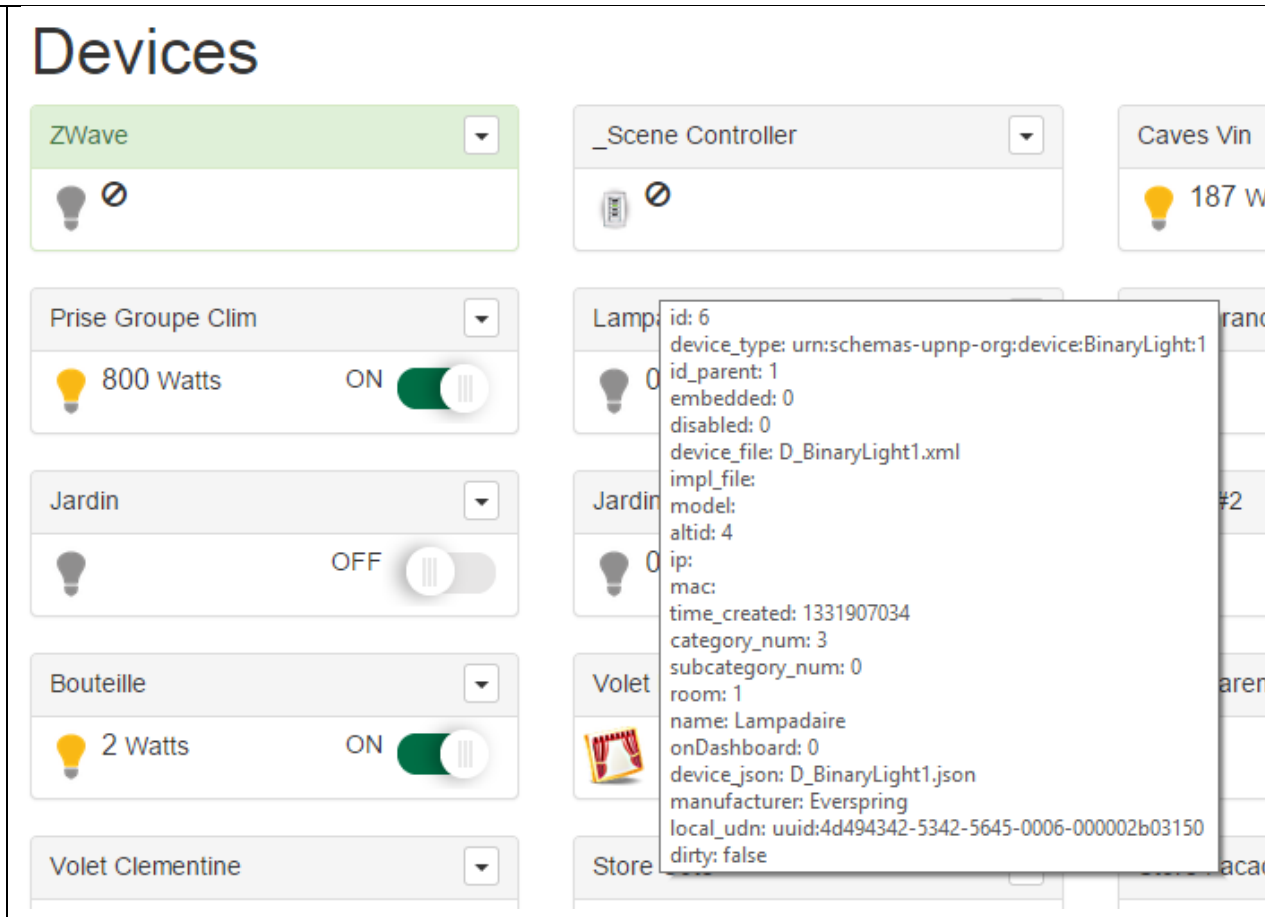
Scene list and execution	

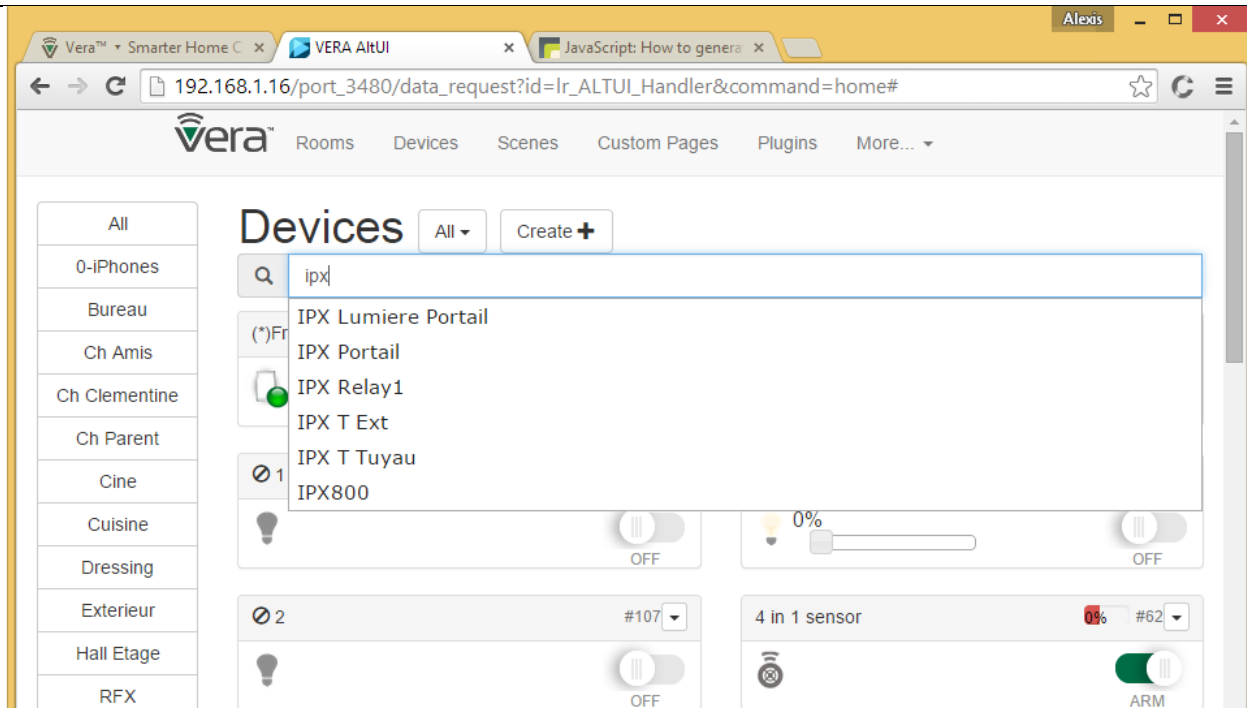
Device Main page

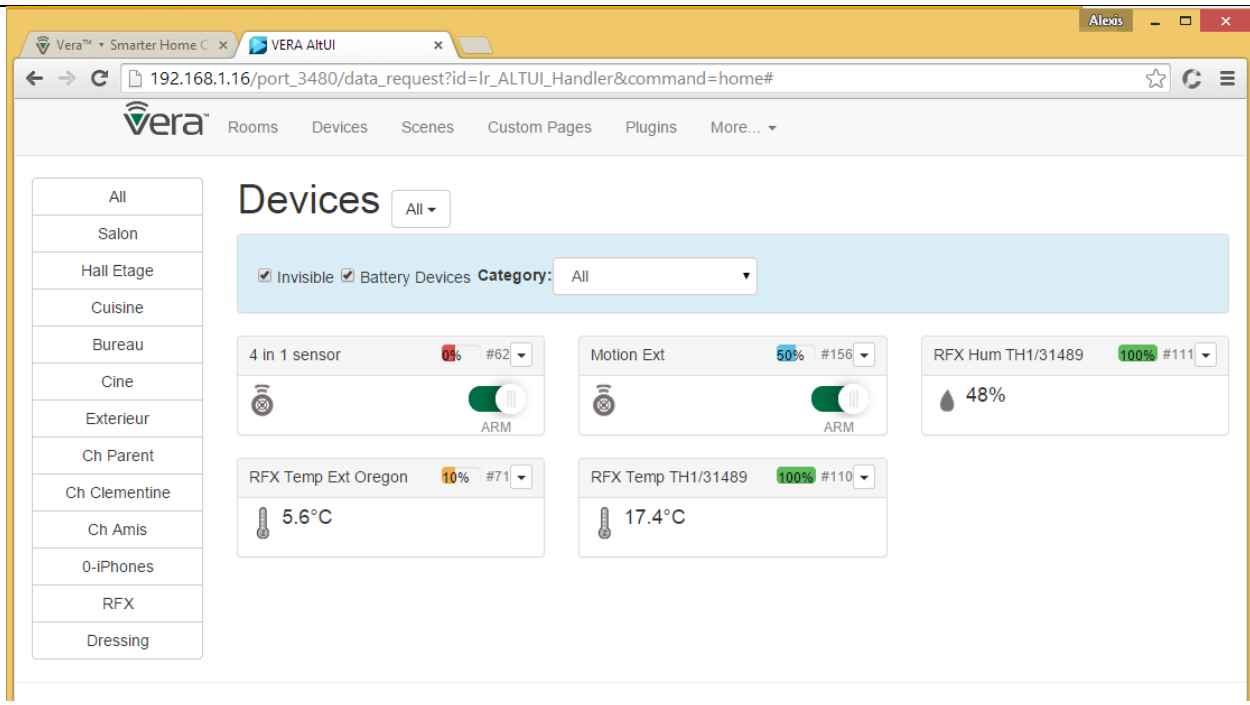
color coding of headers according to device state.

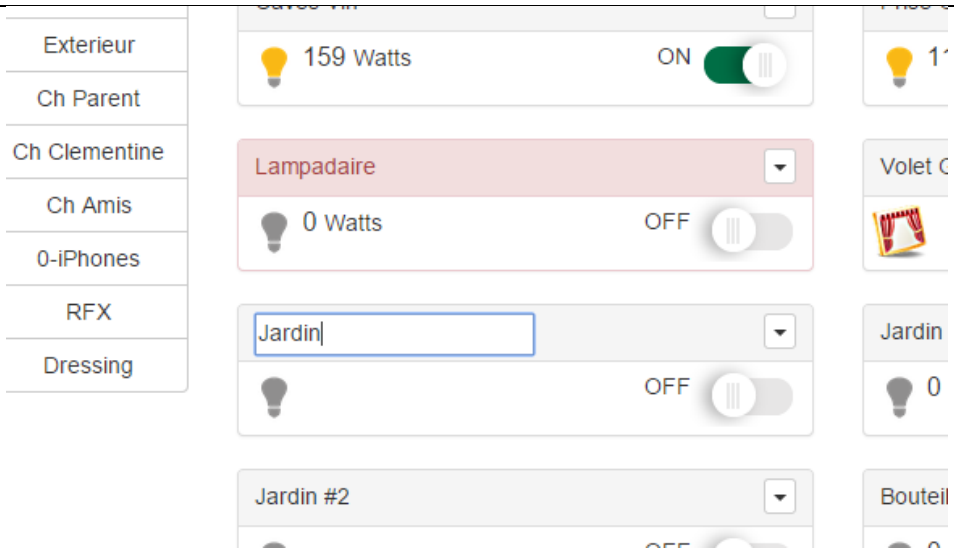
State icons & dynamic display icon logic completely reused from the Vera files JSON description files of vera plugins

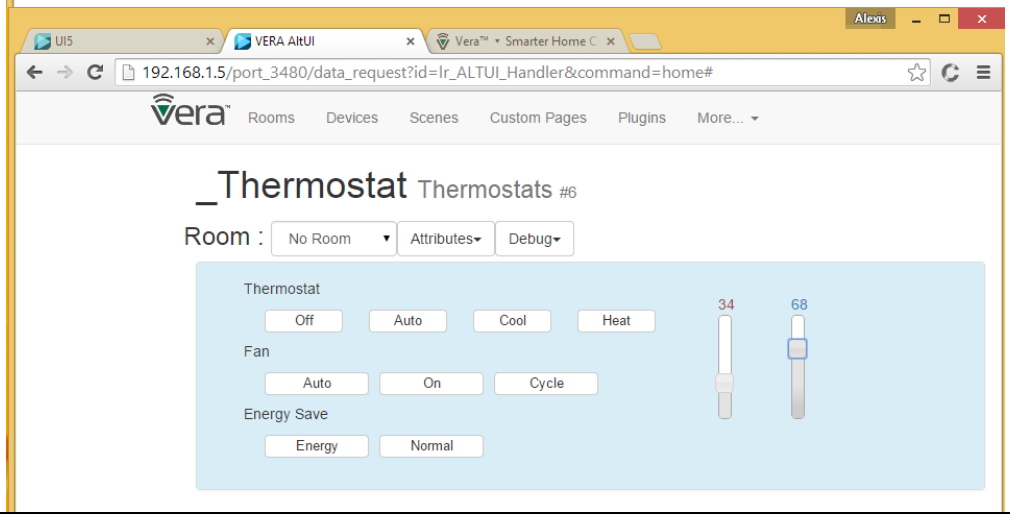


Tooltips with device attributes	

autocomplete filter box	

Ability to filter on device Battery, Display of battery levels	

Click on device title to rename	

Device Control Panel screen emulate VERA and **display the same control panel as the "flash" tab of the device on VERA** Button are functional are trigger UPNP actions.	 

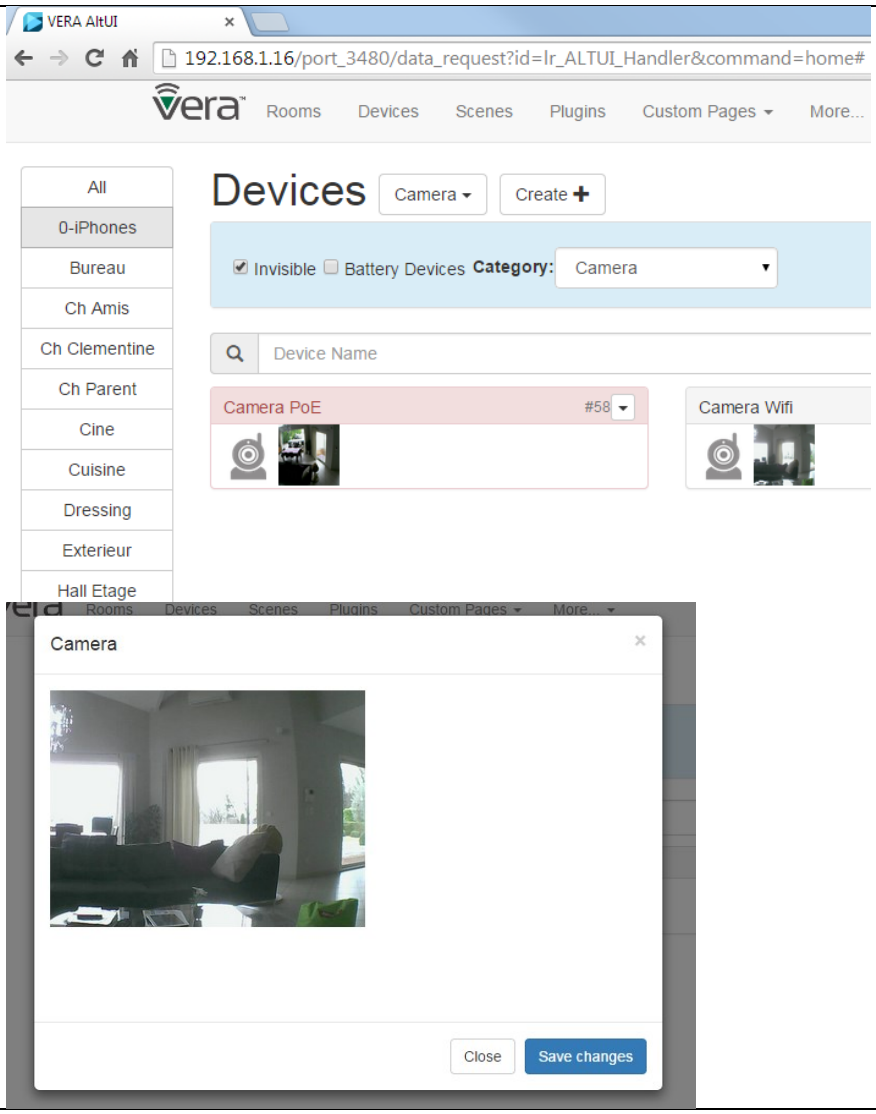
Camera support.

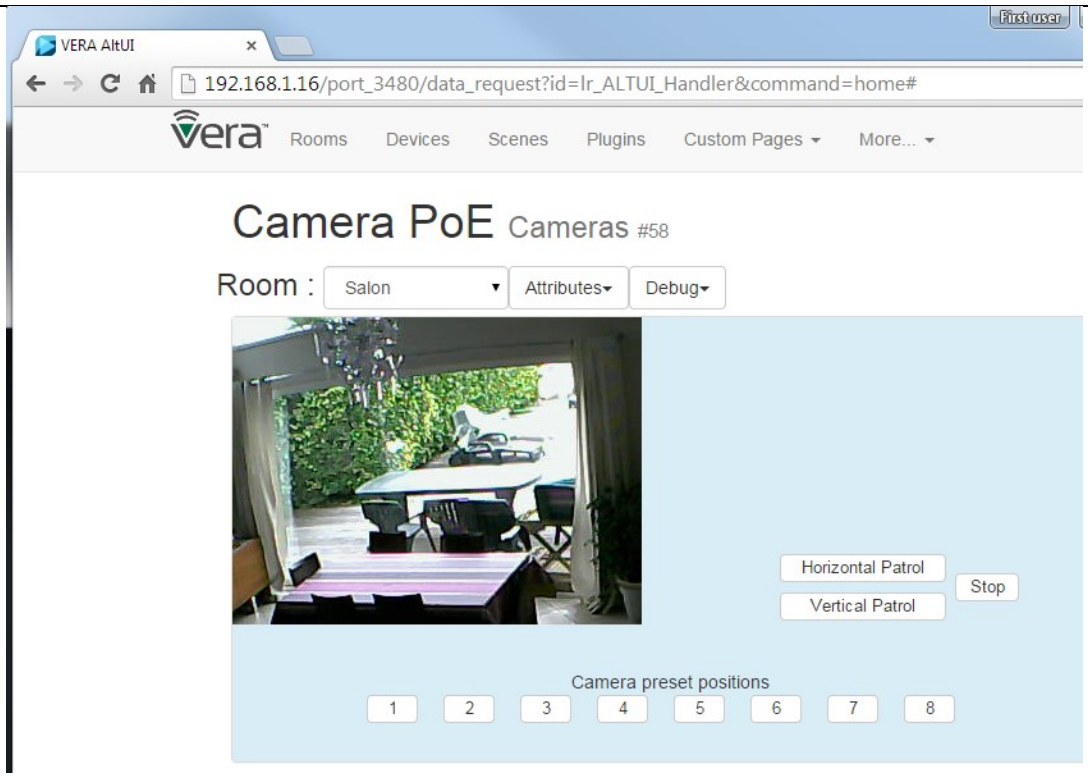
Click on thumbnail to view

Or gointo the device control panel

In REMOTE mode: get snapshot of images

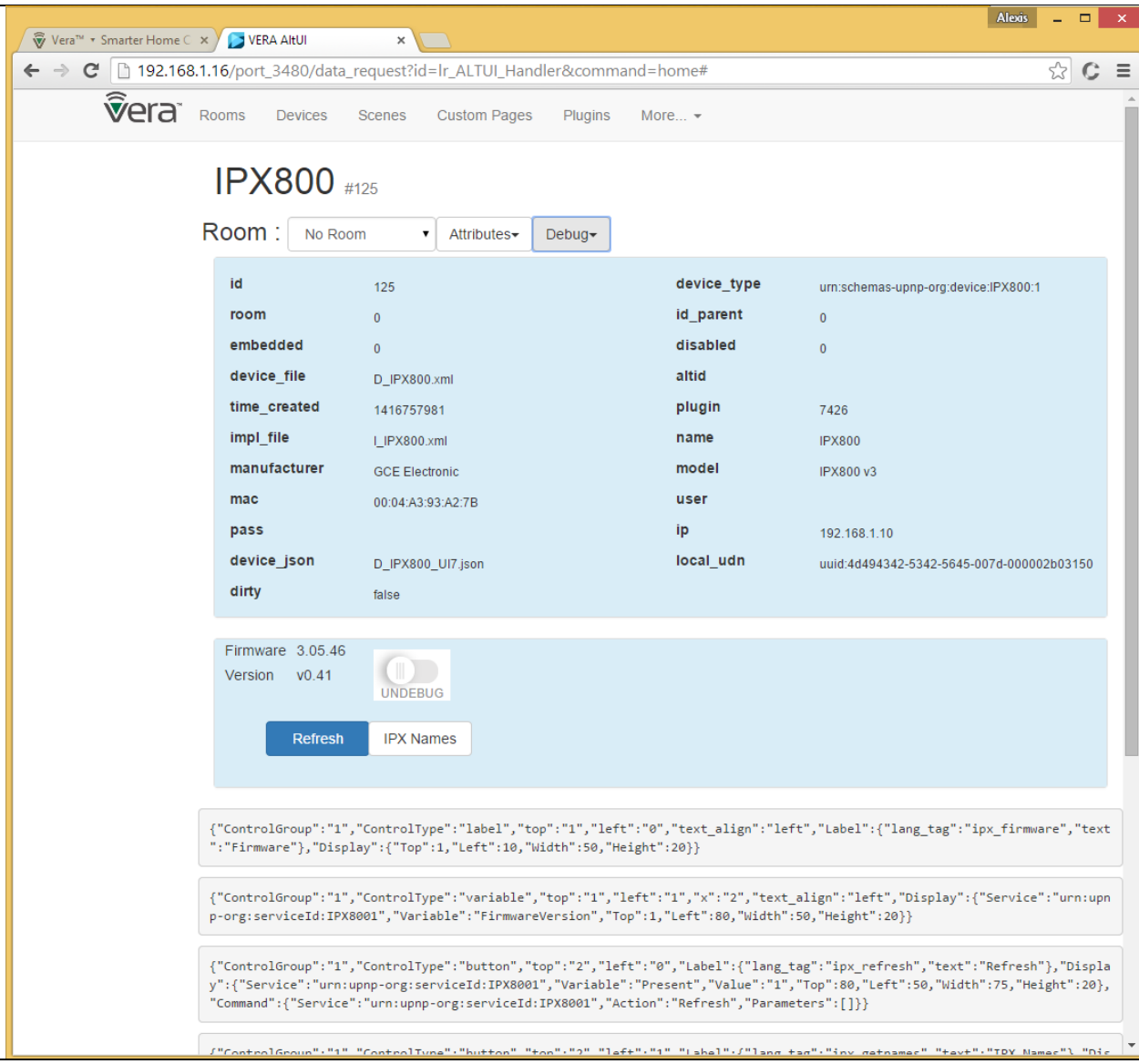
In LOCAL mode: get **direct video** stream

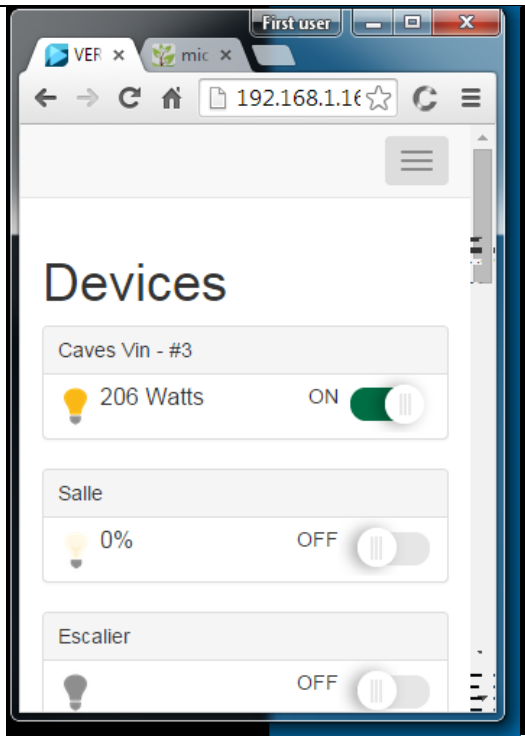
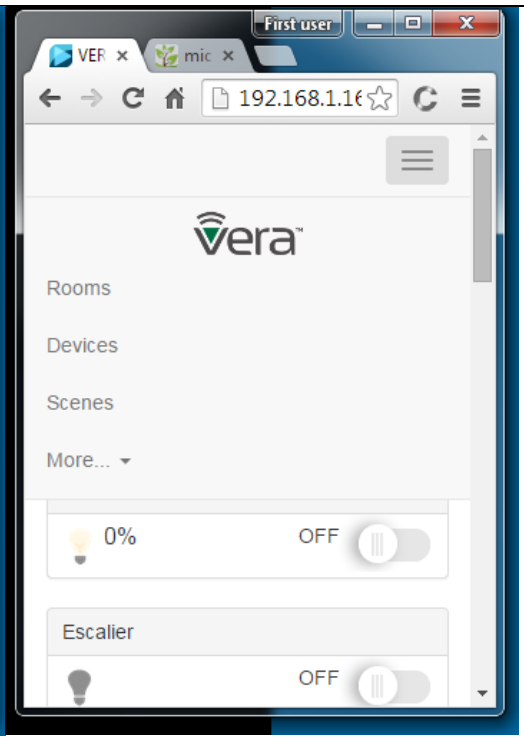


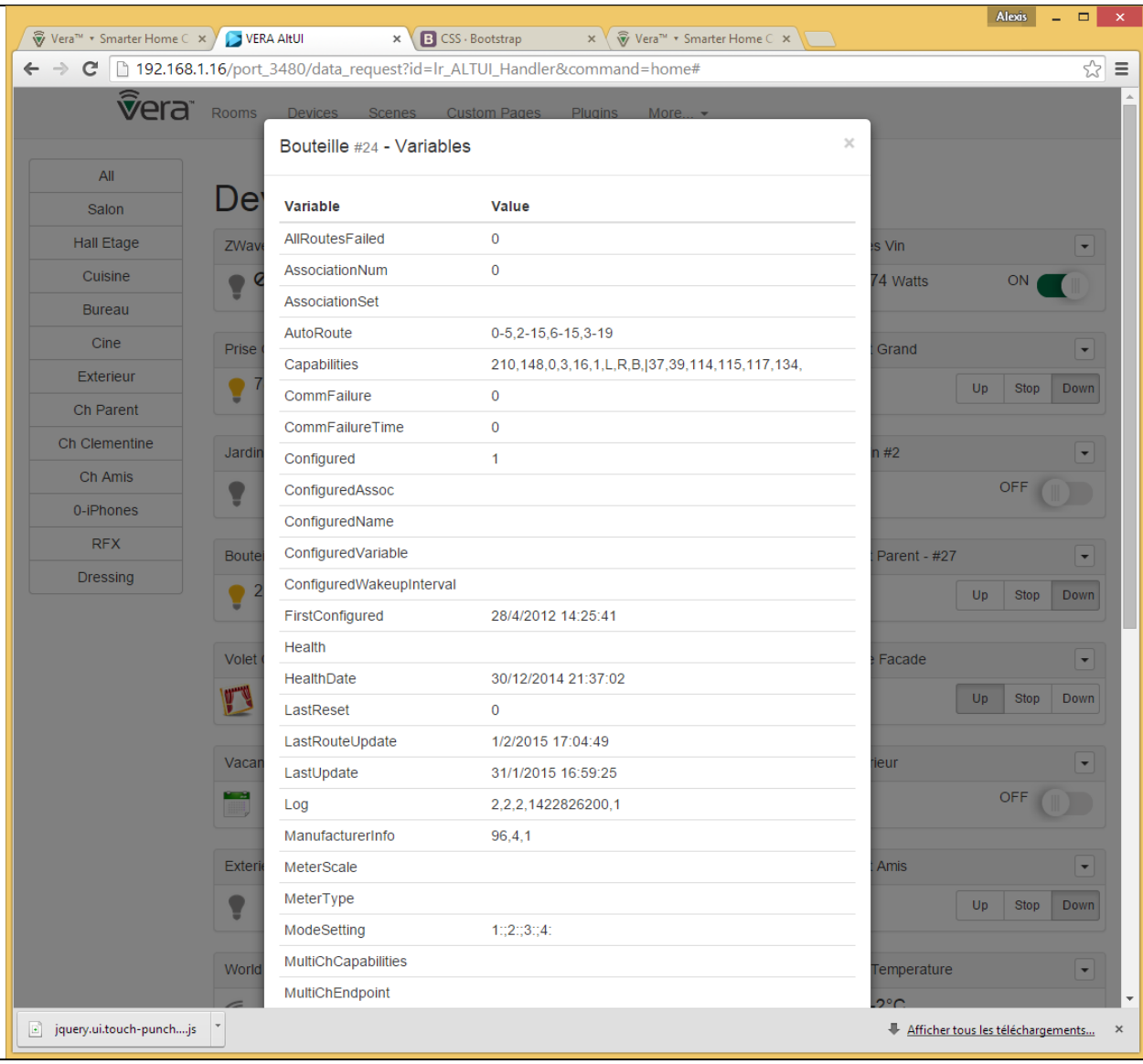

First user
VERA AltUI
192.168.1.16/port_3480/data_request?id=lr_ALTUI_Handler&command=home#
Vera
Rooms
Devices
Scenes
Plugins
Custom Pages
More...
Camera PoE Cameras #58
Room :
Salon
Attributes
Debug
Horizontal Patrol
Vertical Patrol
Stop
Camera preset positions
1
2
3
4
5
6
7
8

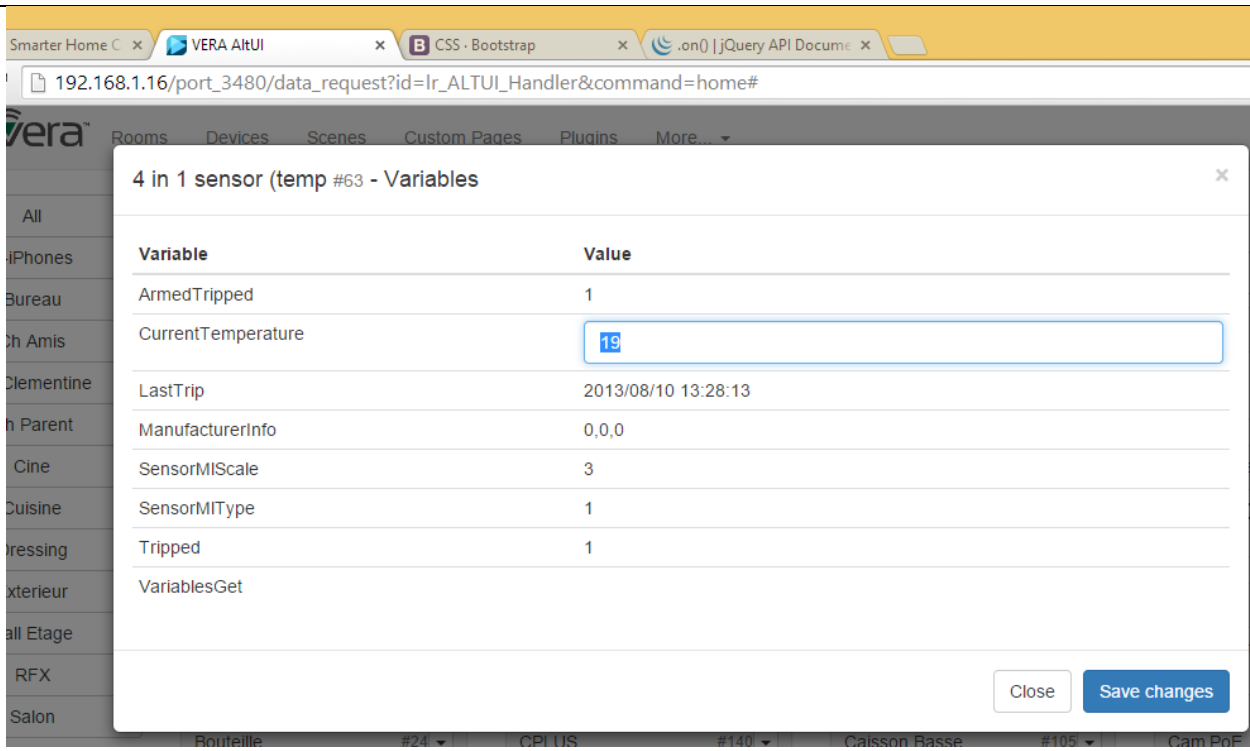
Optional display of device attributes &

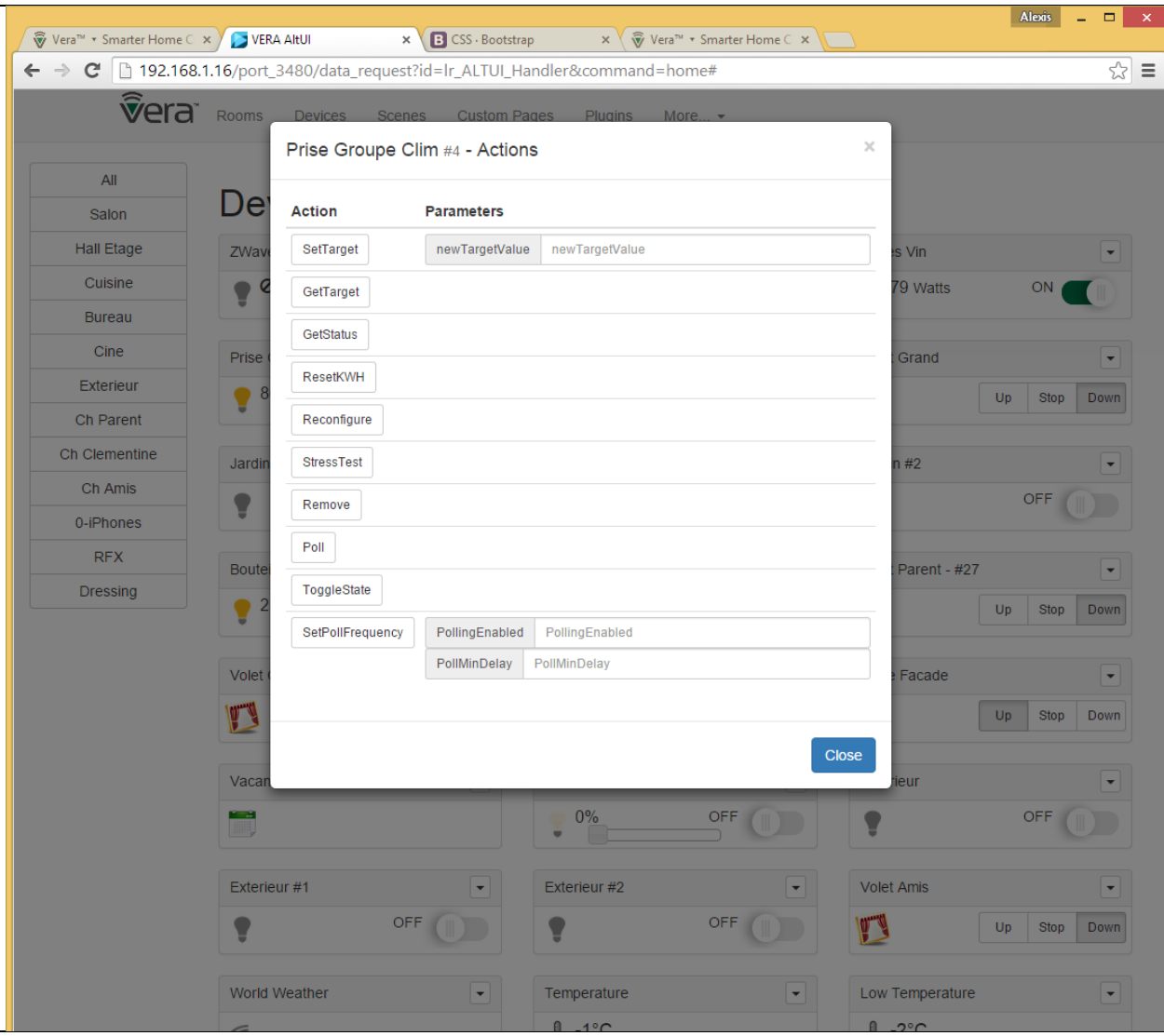
In DEBUG mode only (flag on the LUA device) We can see the Control tab json definition (for debug)

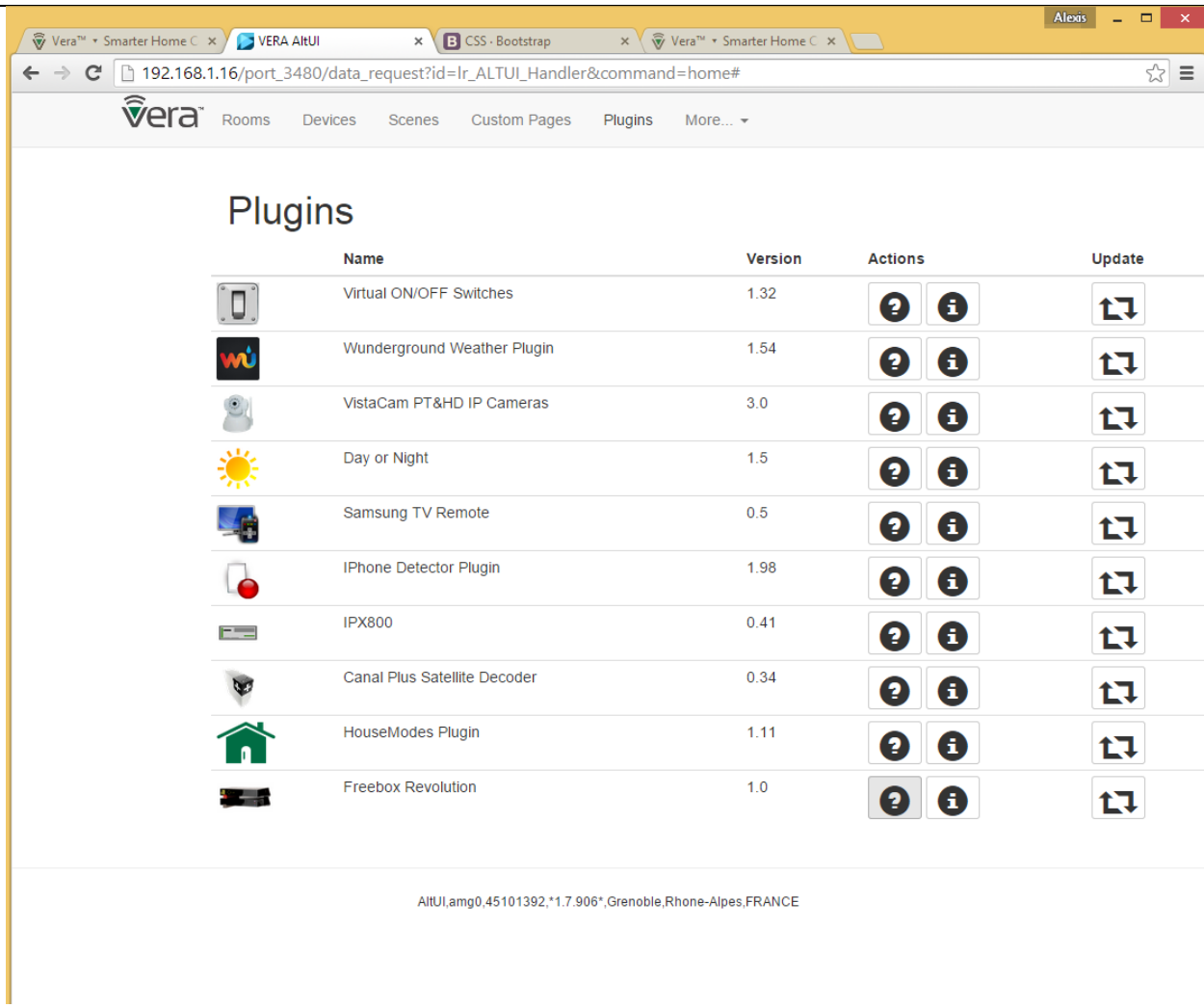


Fully Responsive design that works on iPad or even iPhone 4S small screen On iPad for instance, it adds columns when you rotate the iPad	 

Device variables presented. Timestamp presented as dates	

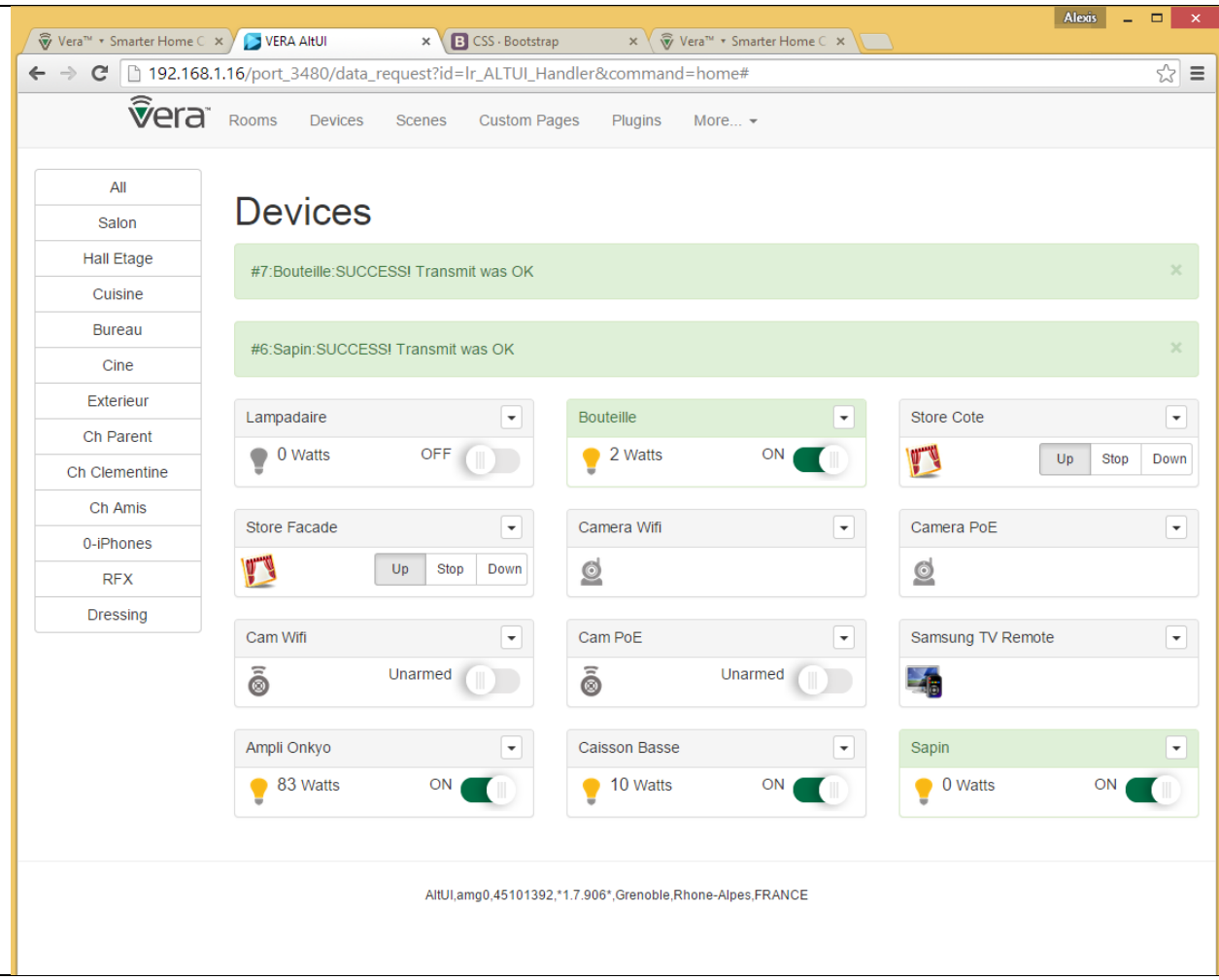
Edit device variable by click into , then click out	

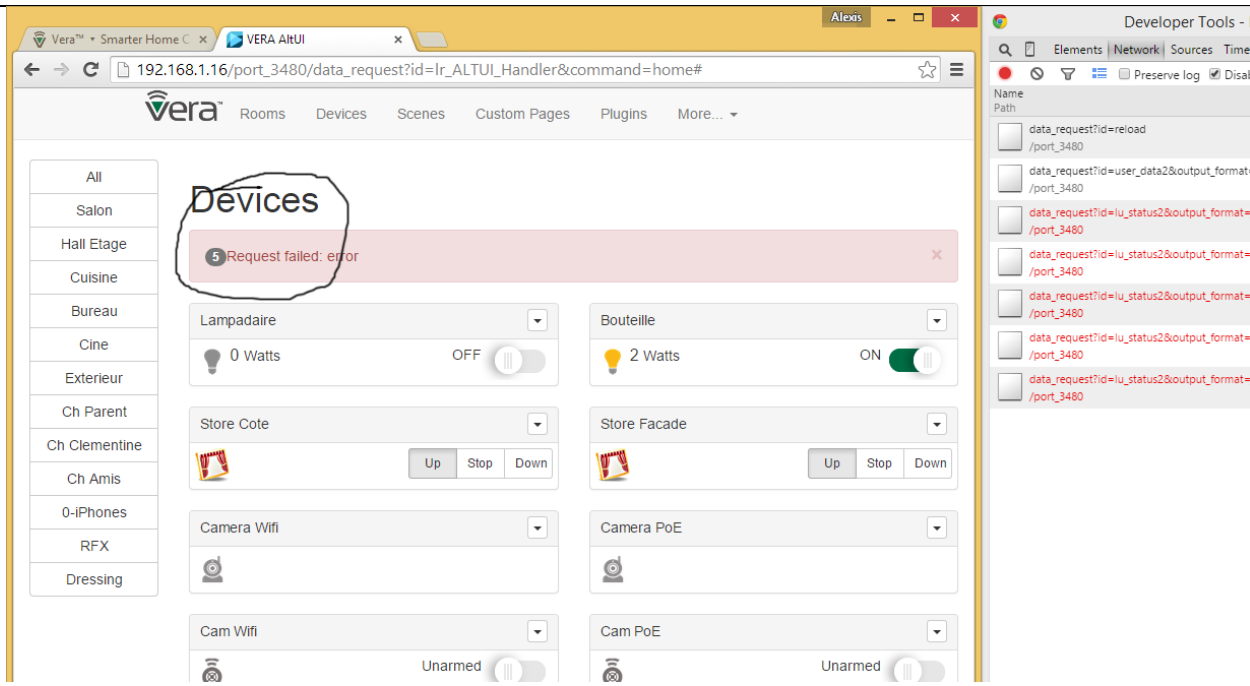
Device UPNP action & parameters callable from the user interface. UPnp definitions dynamically read from the D_ & S_xx files	

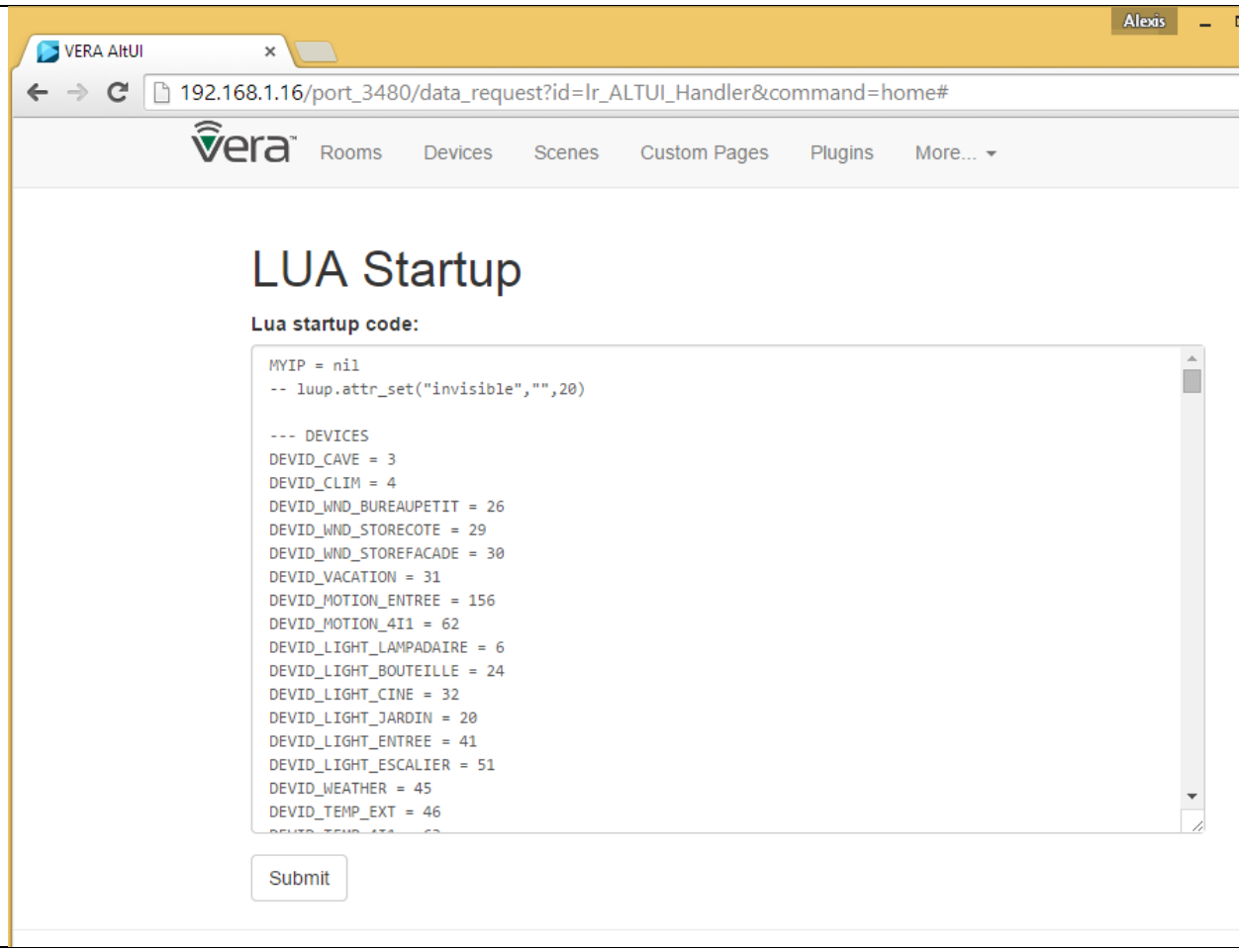
Installed Plugin screen and Update with a button	 Click on “I” directly brings to Apps Store Click on “update” triggers an update of the plugin from the apps store Click on “?” opens the developper HELP page

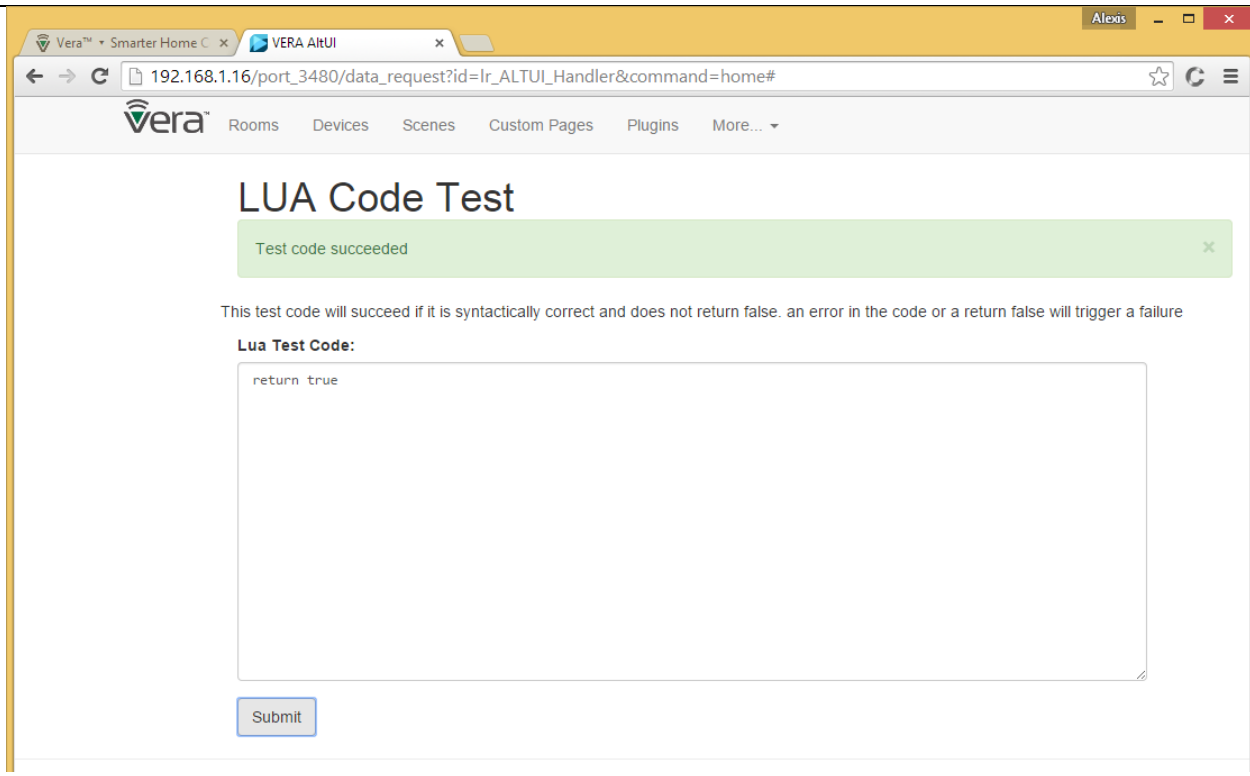
Message Box for messages.

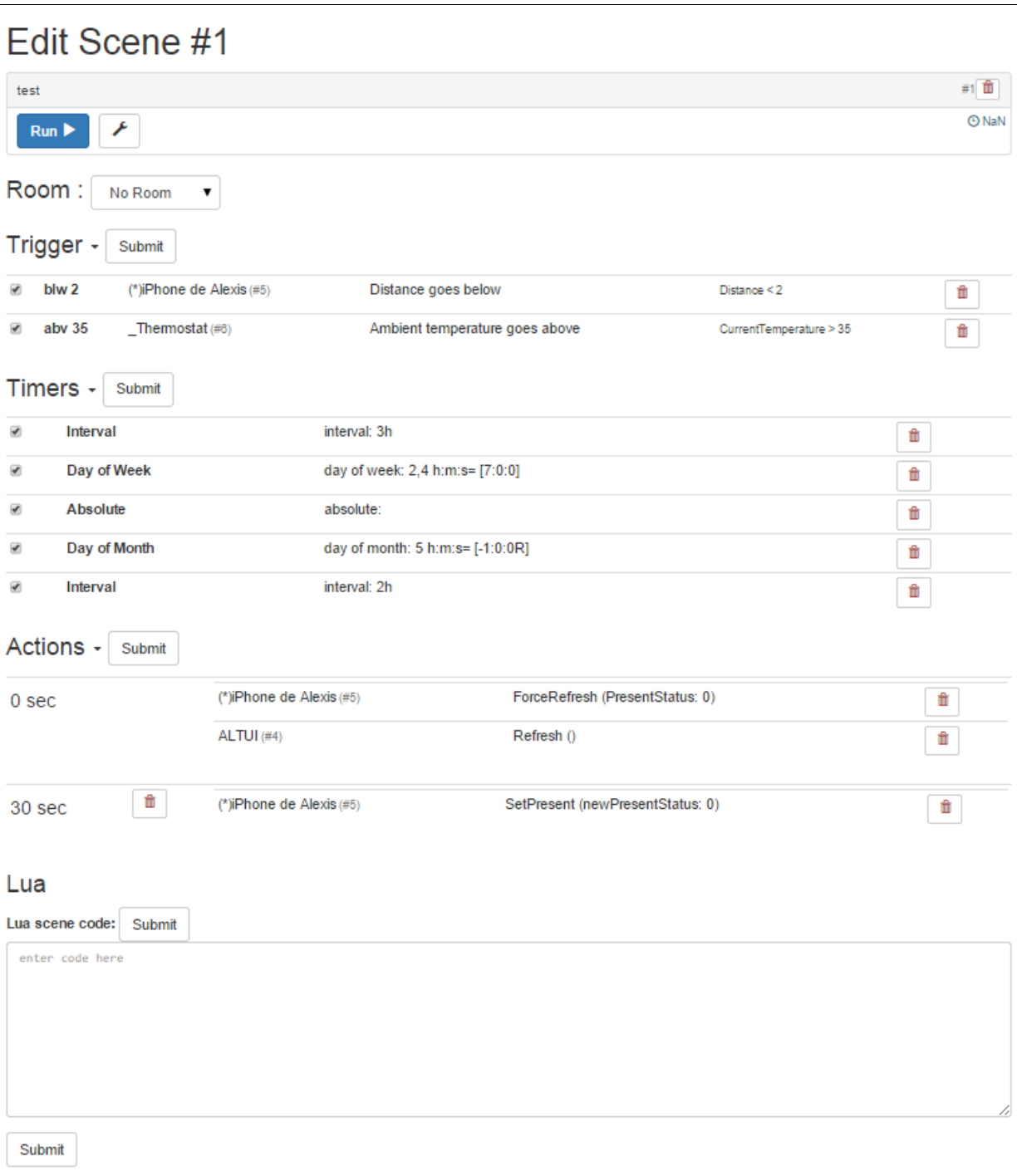
Badge for repeated messages.

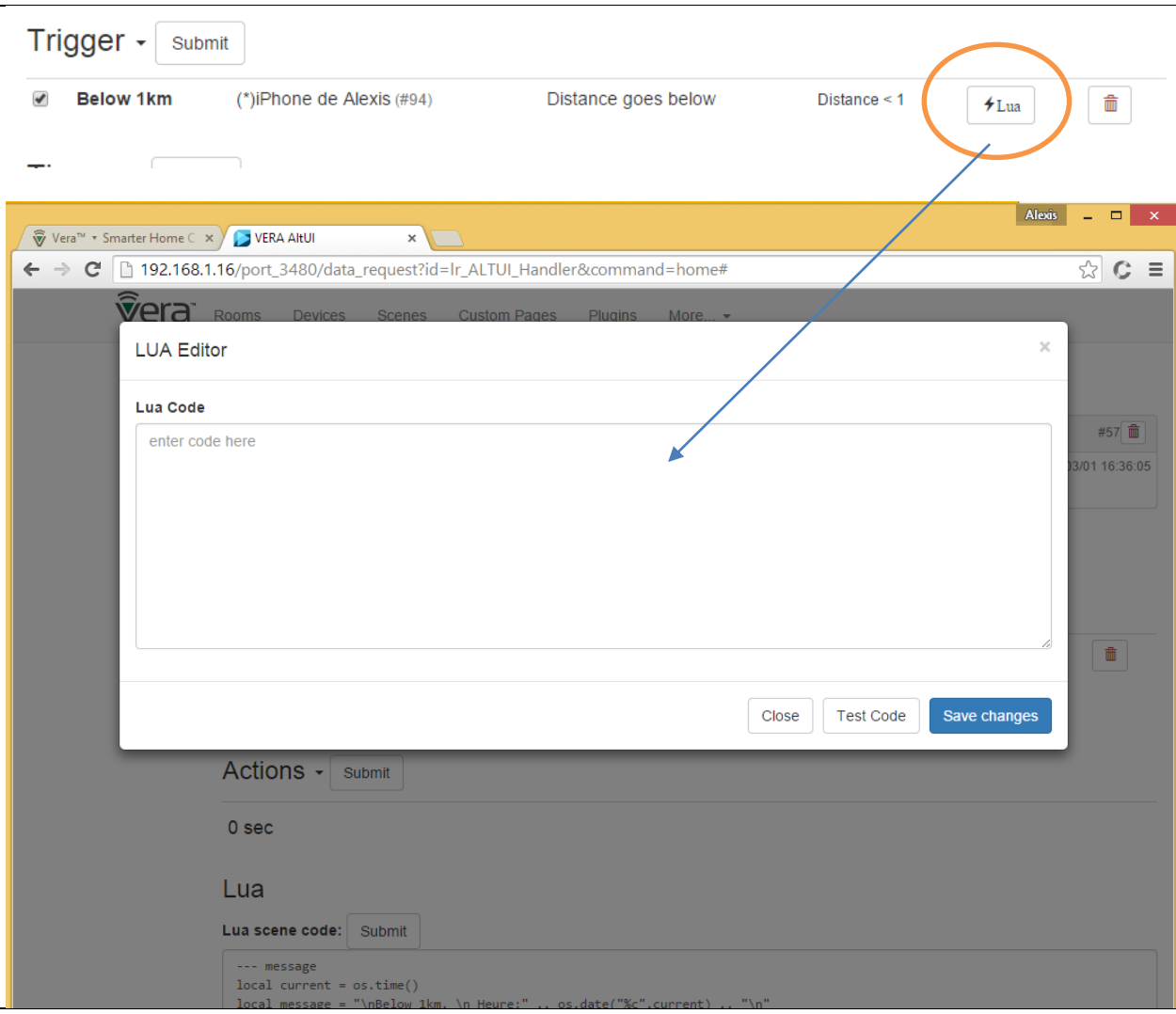


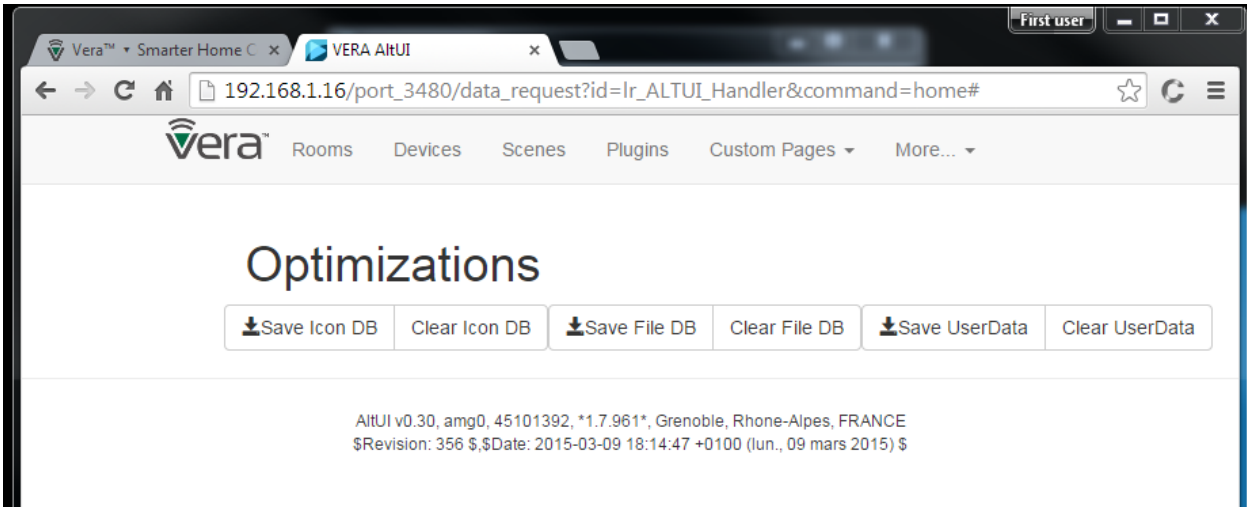
Example of “grouped” error message with a badge number while LUA is restarted for instance	

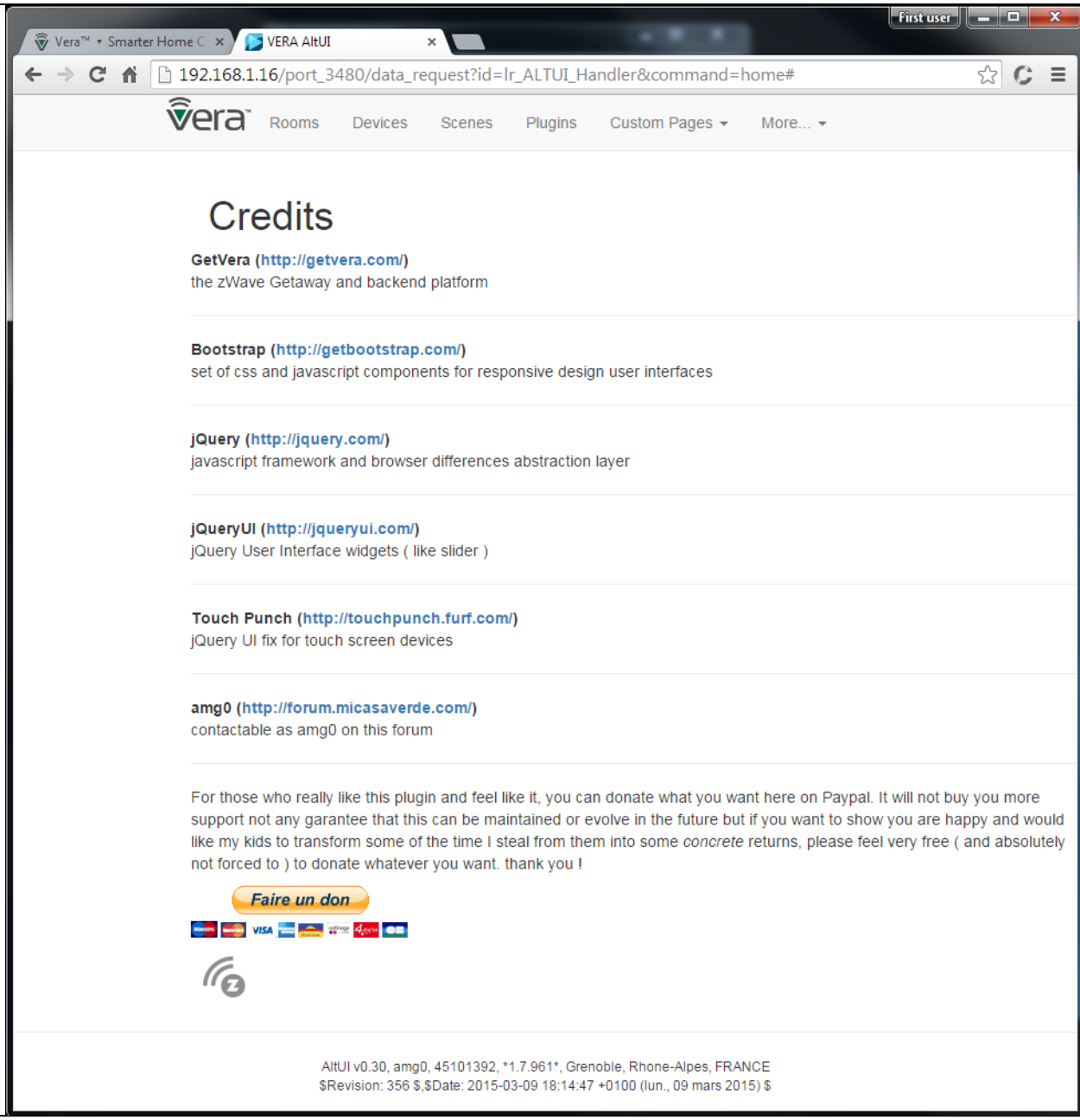
Modify Lua Startup editor	

Lua test code	

Scene editor	

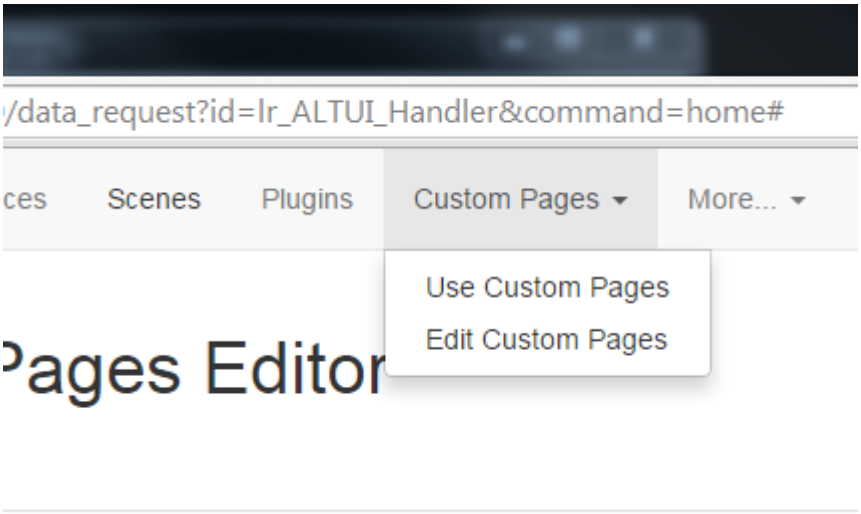
Lua event trigger (does not exist any more in UI7 but it works fine) so I added it back with a test code button right there	

User controllable Cache	Cache for Icons (in remote access, icons are delivered as data uri , base64 and can be cached by the app) Cache for device pnp files (D_xx S_xx) to avoid reloading when not needed. Cache for last user_data to optimize useage from remote location. 

Credits	
Plugins / Custom device	

Custom Pages

The following below explains the concept around custom pages. You can basically create your own panels and retrieve these panels whenever you want. For this you have 2 new Menu commands

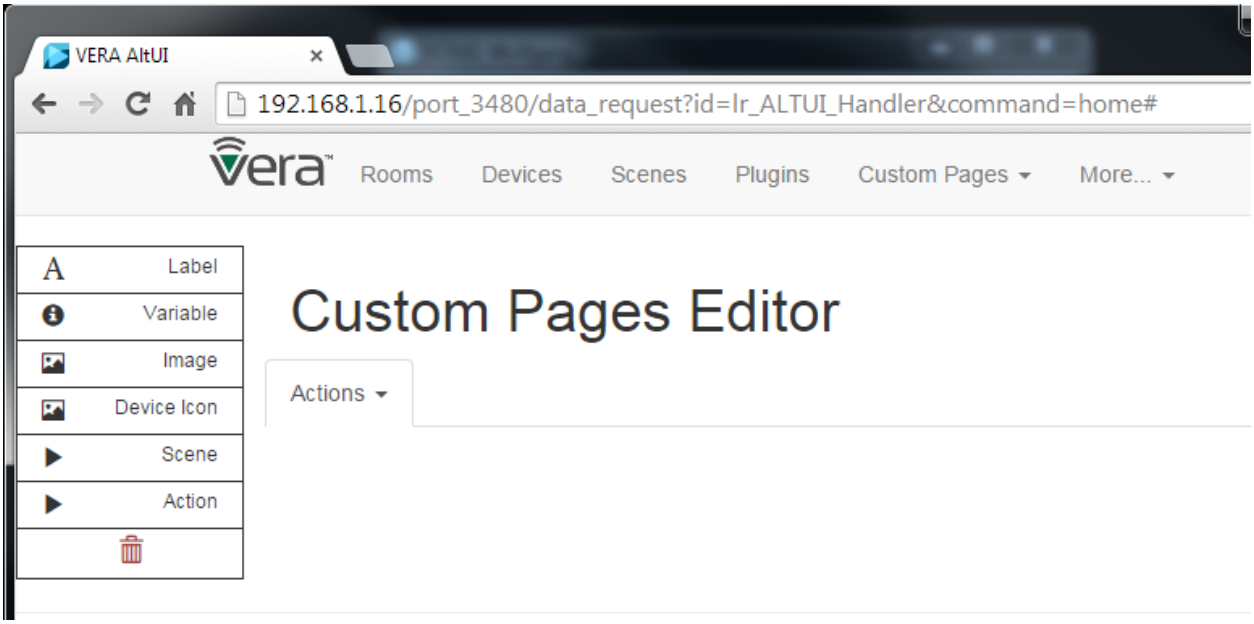


- Use Custom pages : just for readonly use of a custom panel you have built
- Edit Custom pages : to edit the panel.

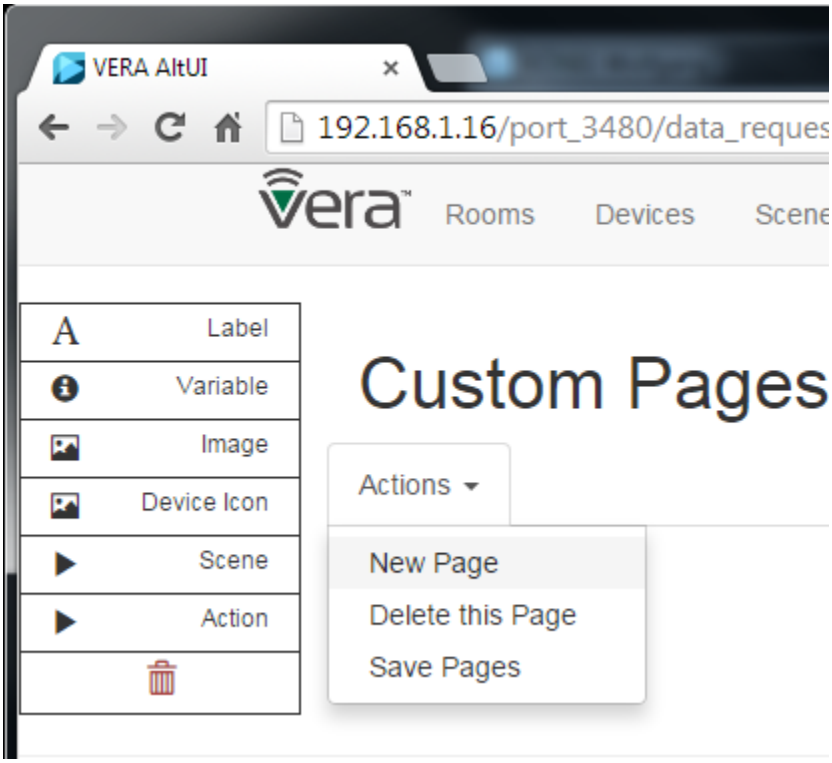
The first time you go there, you will not see any custom pages as you have not created any yet. So let's start by going into Edit mode first.

On the left, you have a list of tools in a toolbox. For now there are 3 tools:

- The Label one : to show a static label
- The Variable one : to display a current device variable value
- The trashcan : to delete a widget from a panel screen by drag and drop.

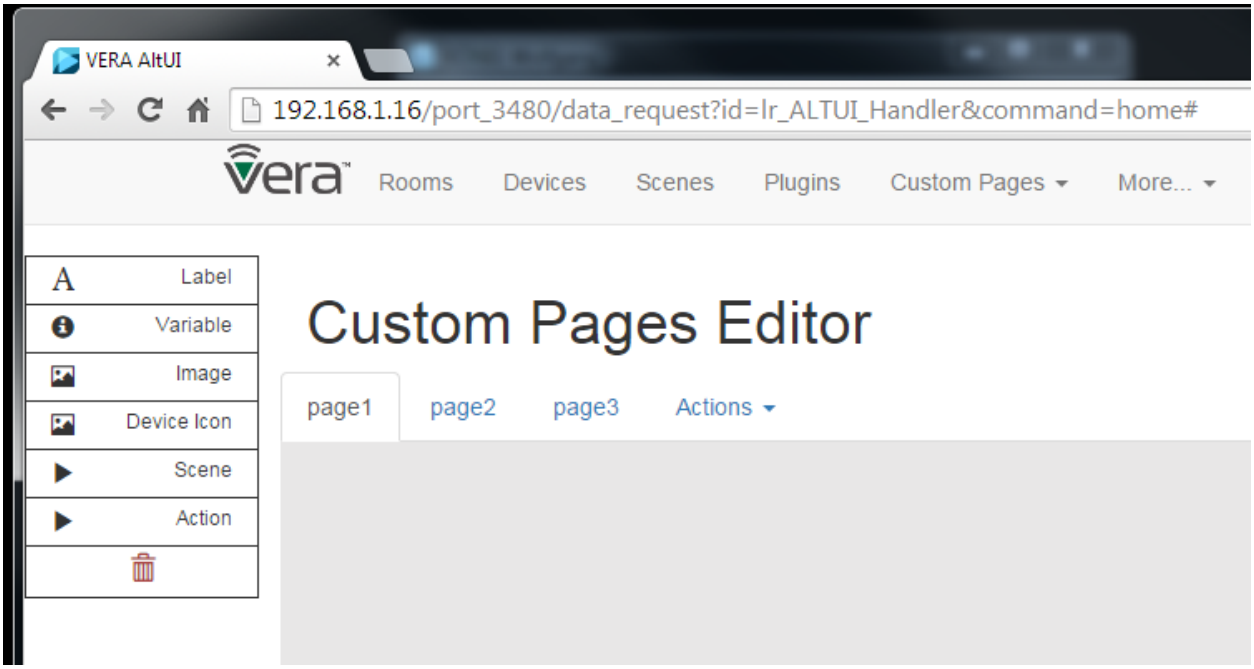


For now, you do not see any page , so let's create one by going into the menu Actions.

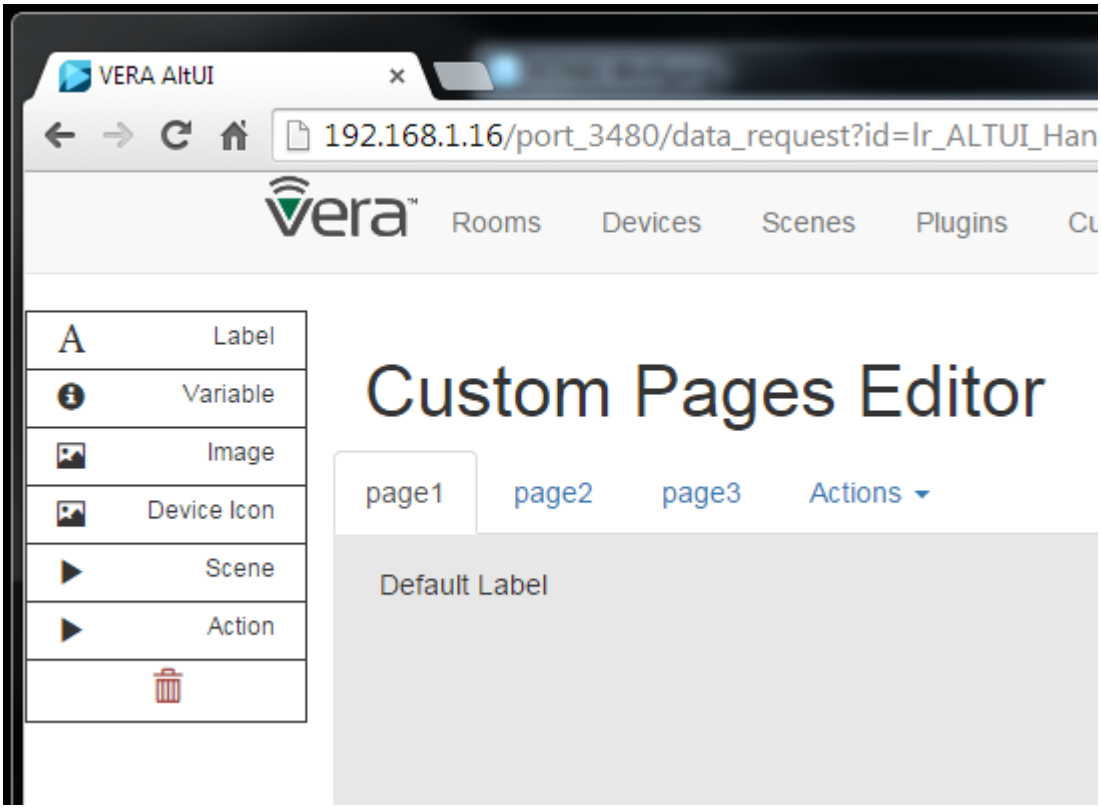


Click New page and your first page is created and is empty for now, but you see a grey canvas where you are going to position your controls.

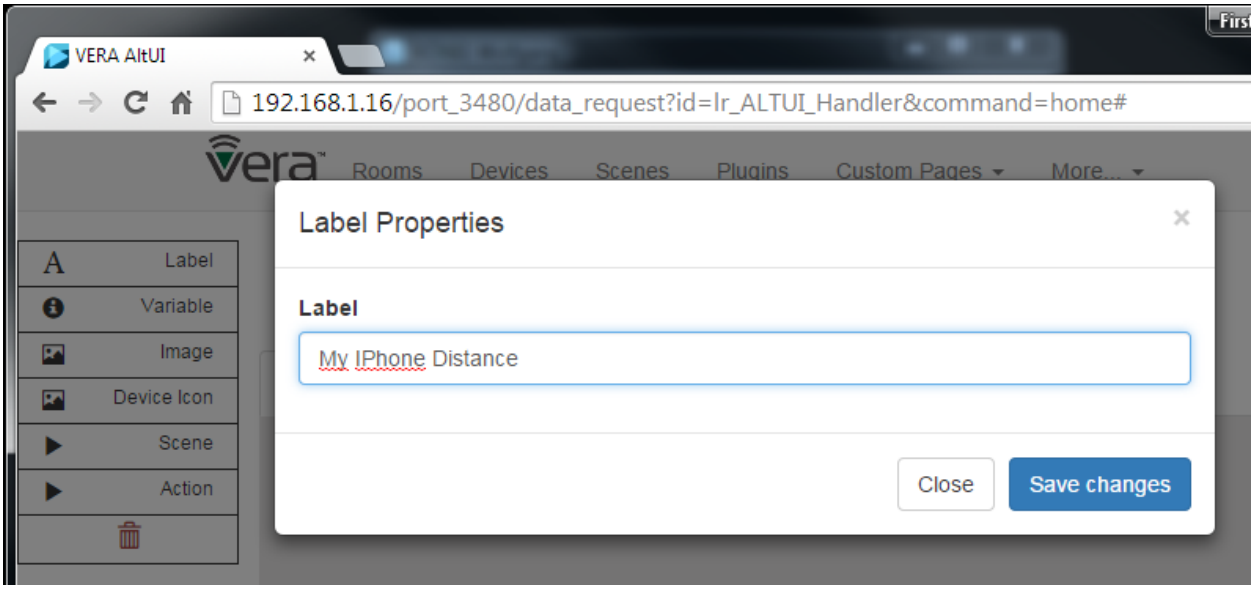
You can create several pages, they will be displayed as "Tabs" you can select to move from one page to the other.



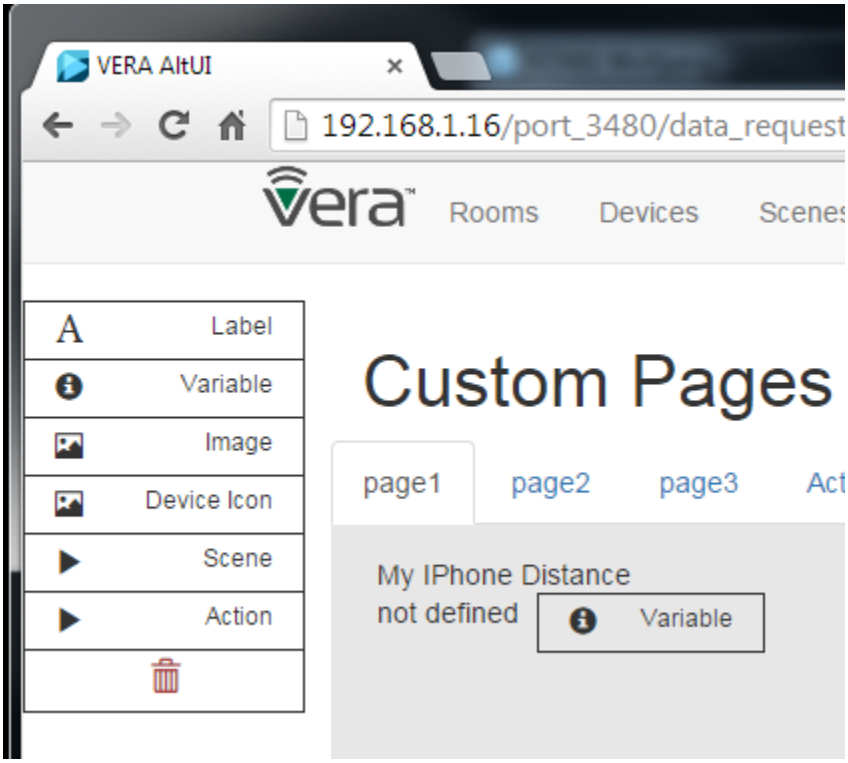
Now, lets position a few control on the panel. I have a IPhoneLocator plugin in my vera and I want to display the distance and the unit it is reporting. So you are going to select the first tool (the A for labels) and drag it into the canvas area. It will create a default label right at the position you left it.



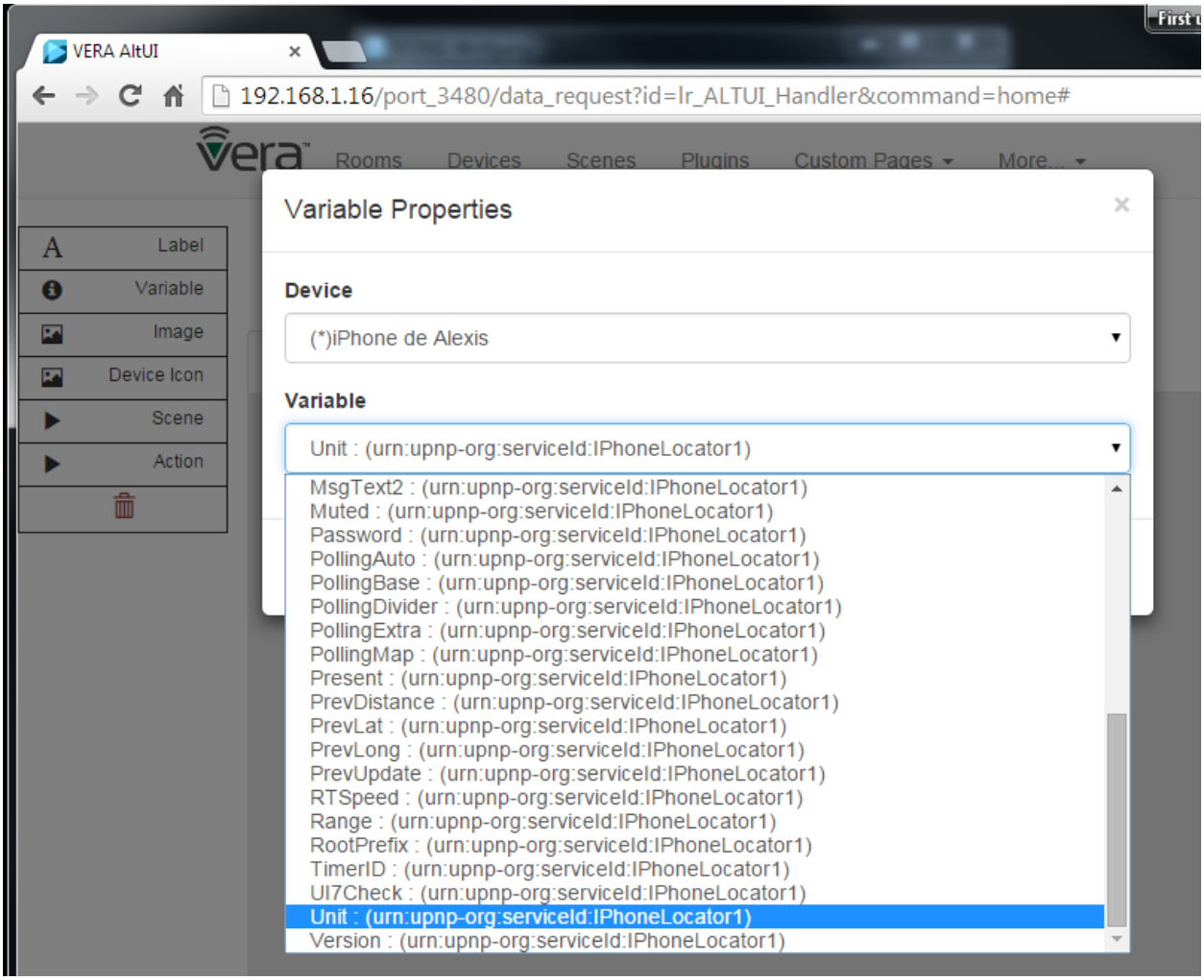
By clicking on it you can change this Default Label.



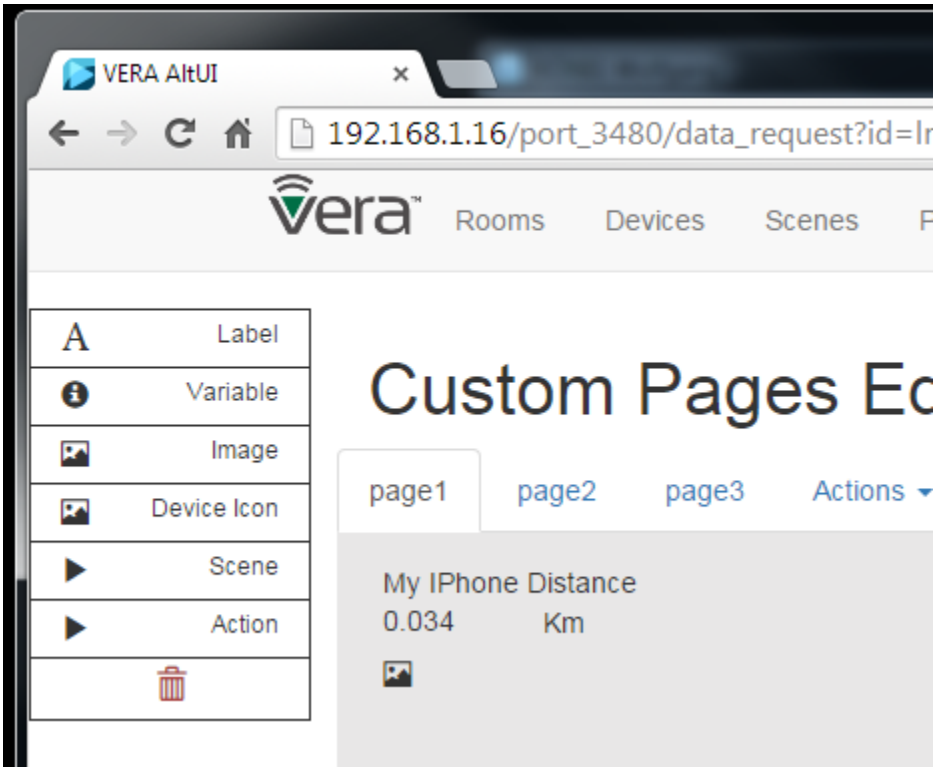
As expected the label has changed on the canvas. I now want to report a dynamic value coming from the device variable, I will use the second tool from the toolbox (the I for Info, which is a variable). I want the distance and the unit which are 2 different variables on this plugin so I will drag and drop 2 “Info” controls. You can move around a control after you have dropped it on the canva surface, just move them around as you want.

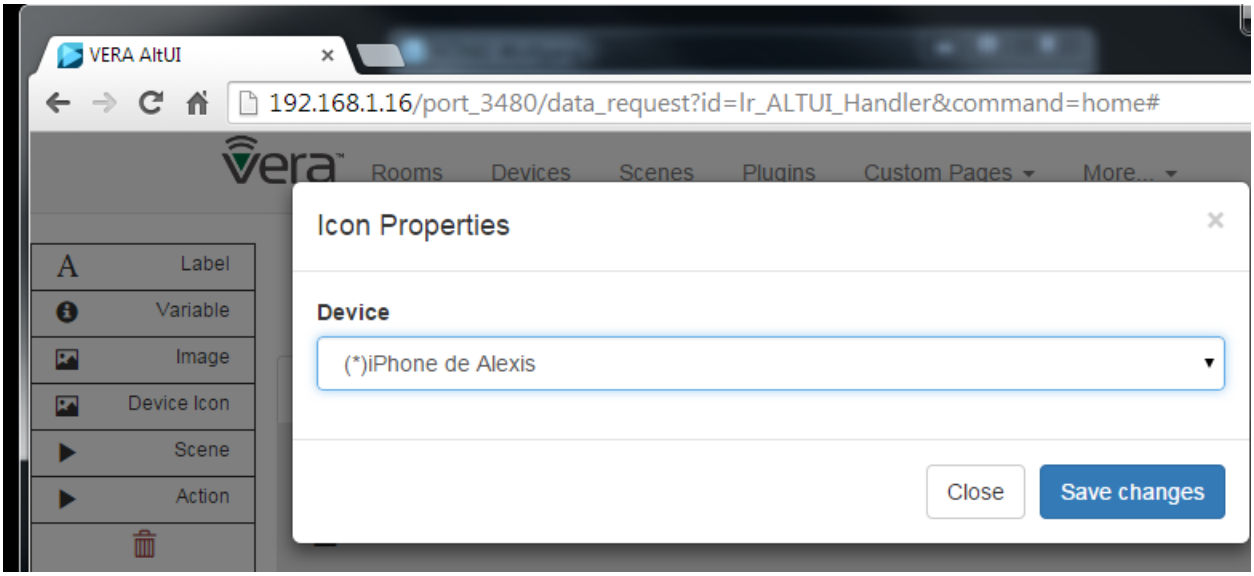


Double clicking on the variable, you can change the parameters so let's now select the right variables.

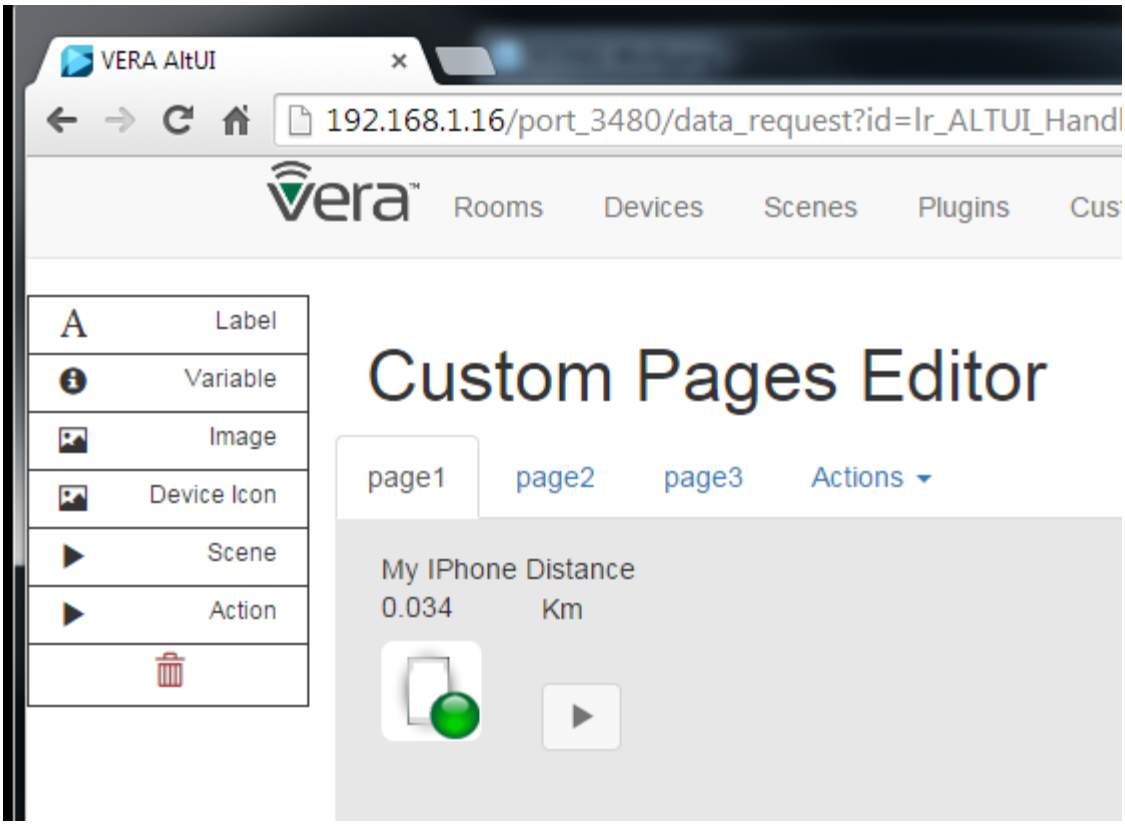


Let's add the icon of the device (which will follow the dynamic states as defined per the plugin author)

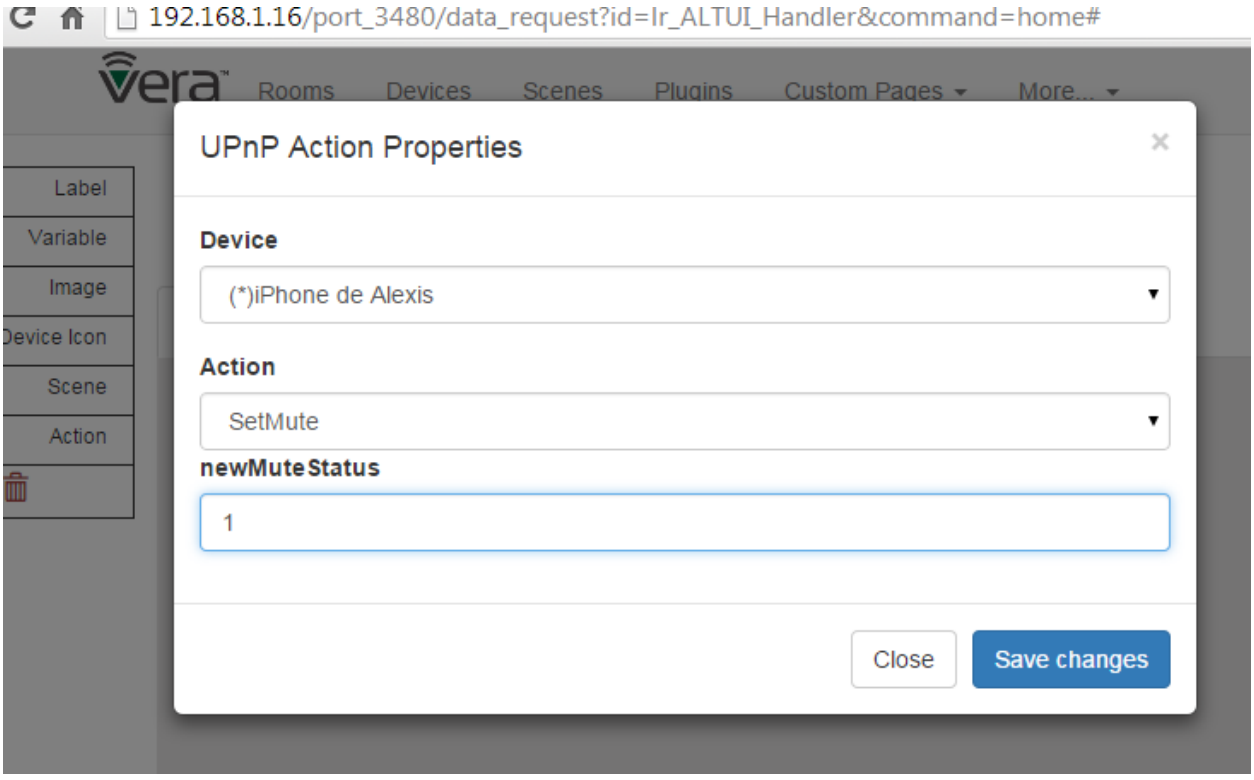




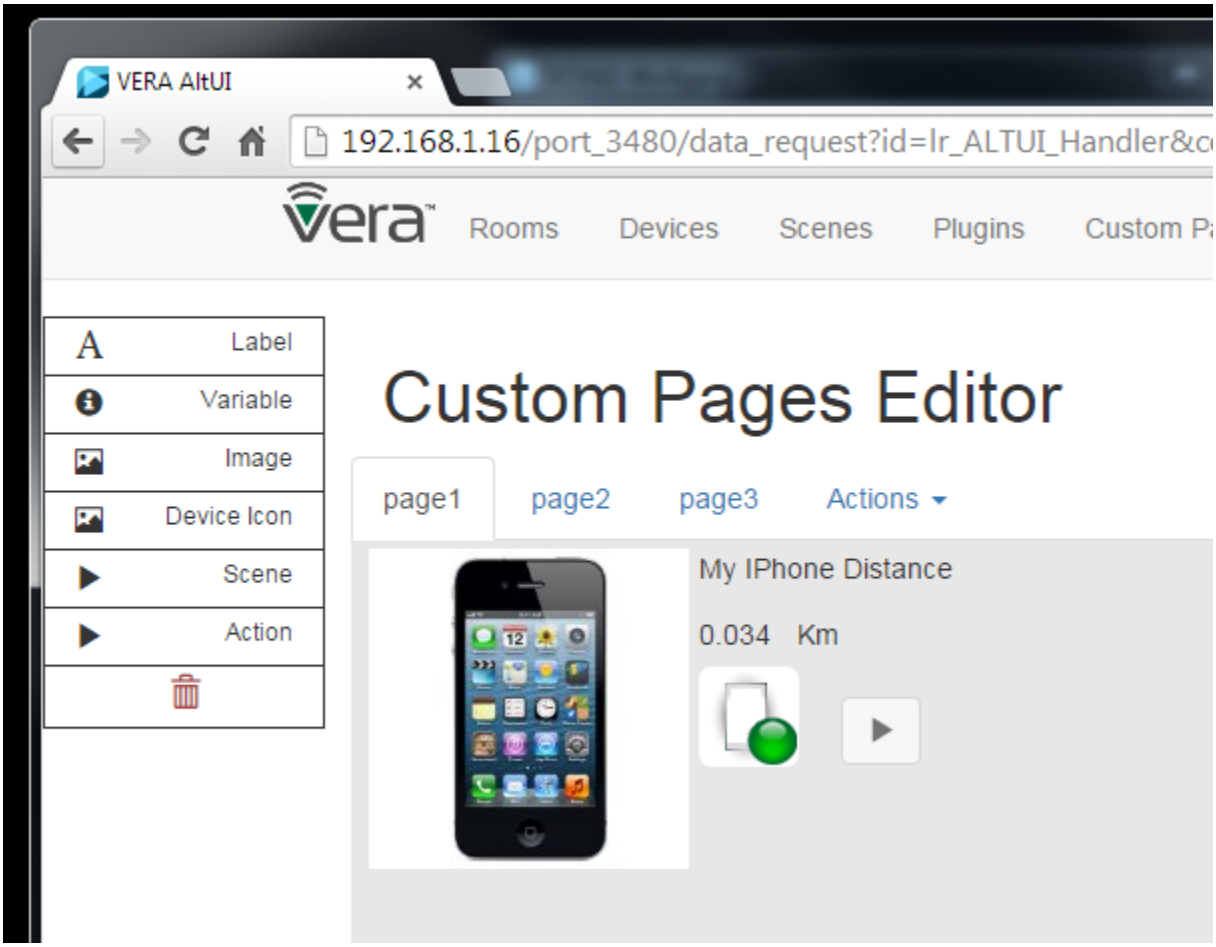
And Let's add a mute button.



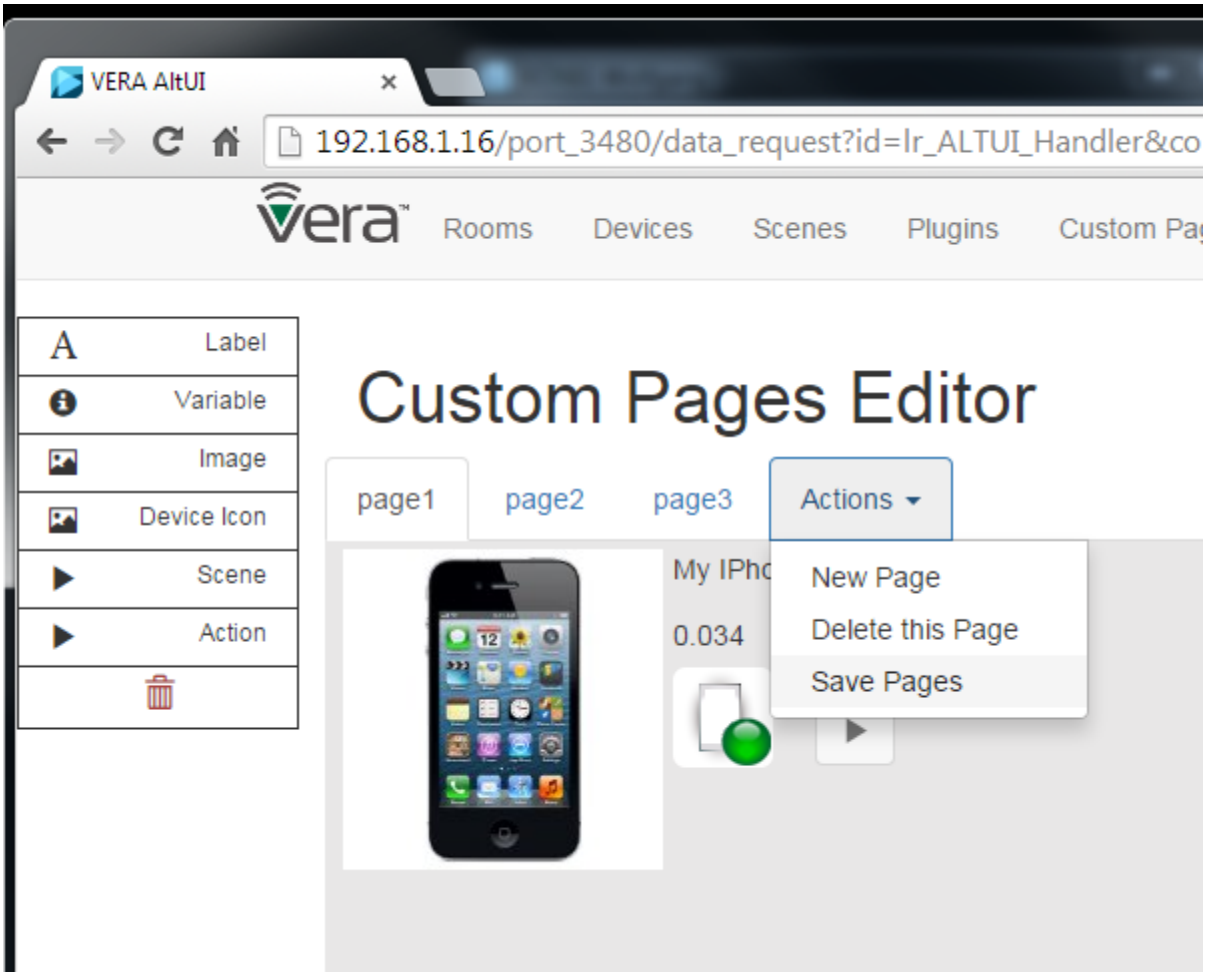
Which we need to configure to run the right UPNP action:



Et voila (with a 3rd tool from the toolbox , image which can be any URL or data uri (for embedded image))

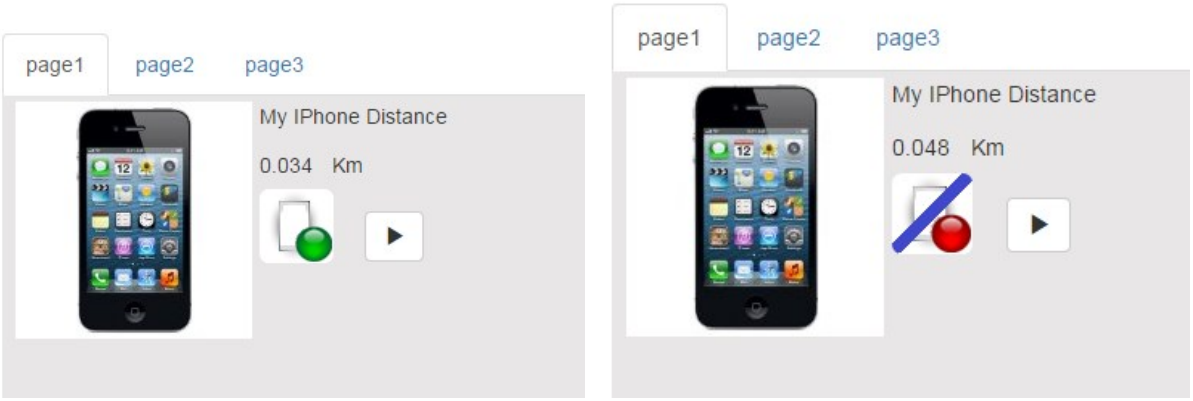


I now want to save it so that it can be persisted and reopened next time so I go into the Actions/Save menu.



That is it , now the page is visible by the Custom Pages / “Use custom page” menu and you can close your browser and reopen it , it will still be there.

Now I can simply use it in read only mode and the button & icon are functional



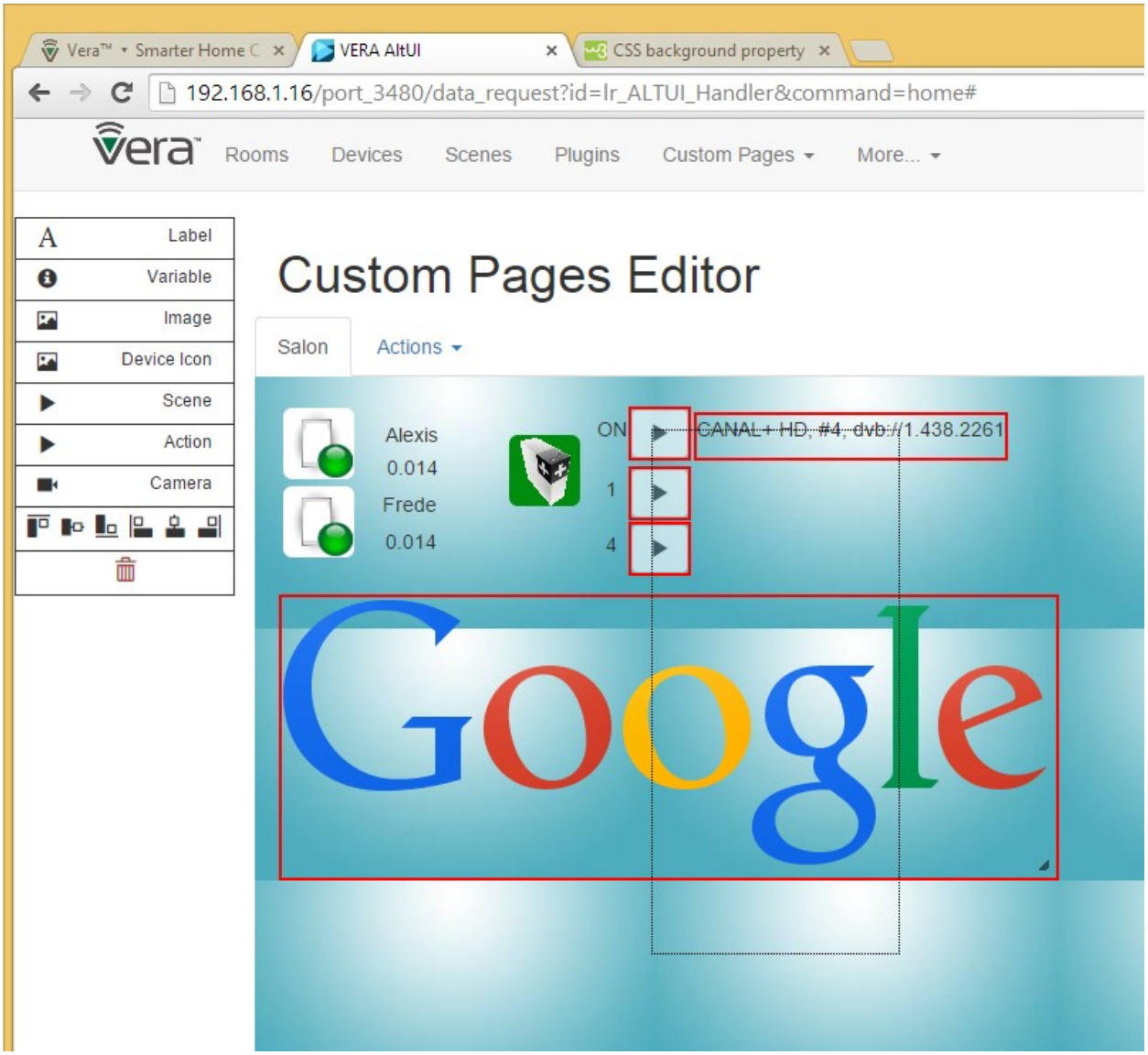
All pages definitions are stored in the LUA plugin variable “CustomPages”, you can see it from ALTUI and copy paste in a JSON online viewer if you are interested

Other capabilities:

- Page Property menu items enables to:
  - Change a page name
  - Change a page background , any valid CSS3 background string is accepted. Solid color, grade, radiants, stripes, url('http://xxxx/image.png') are valid. See the syntax of "**background"** css property

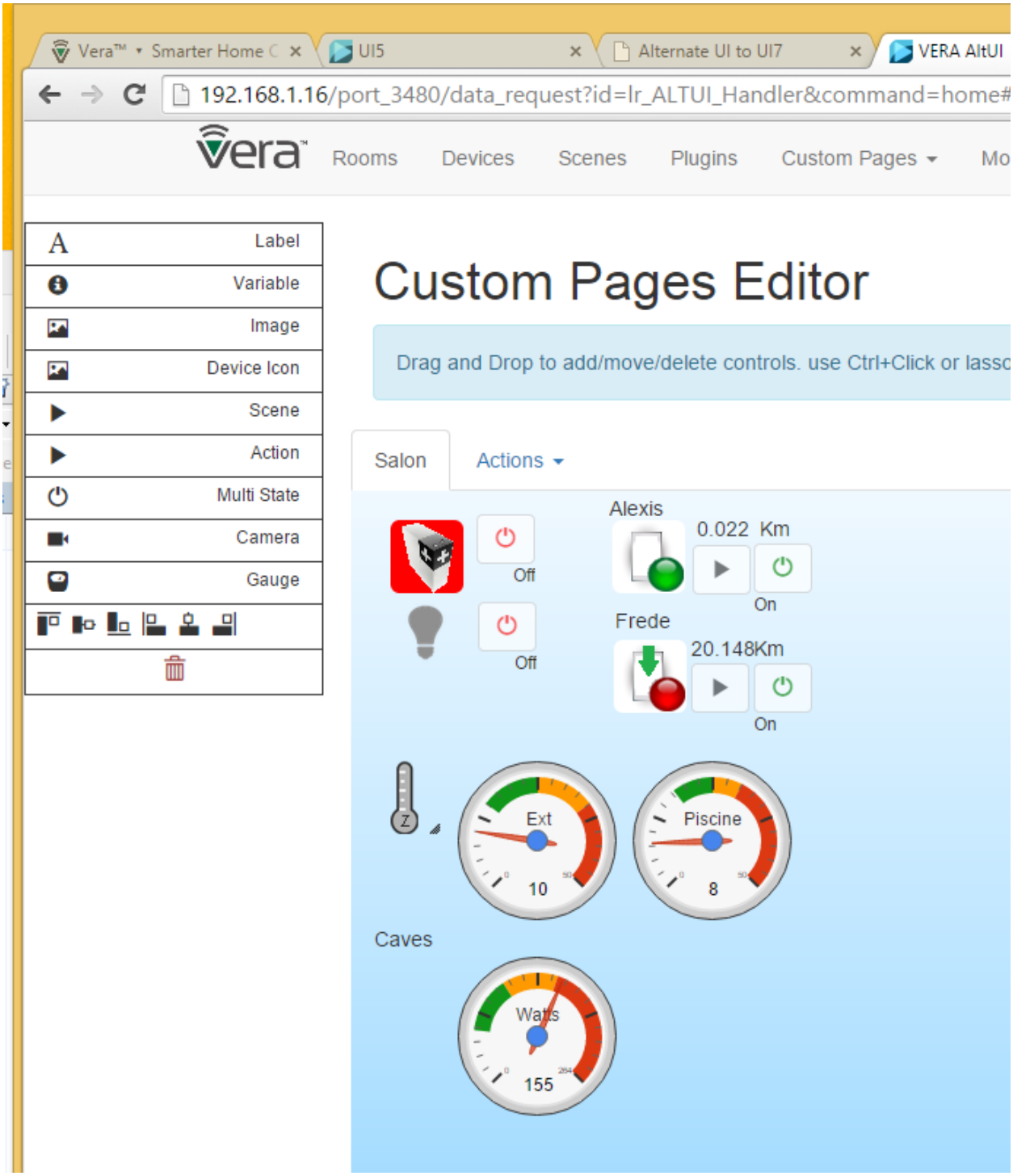
On this Picture you can see various important elements:

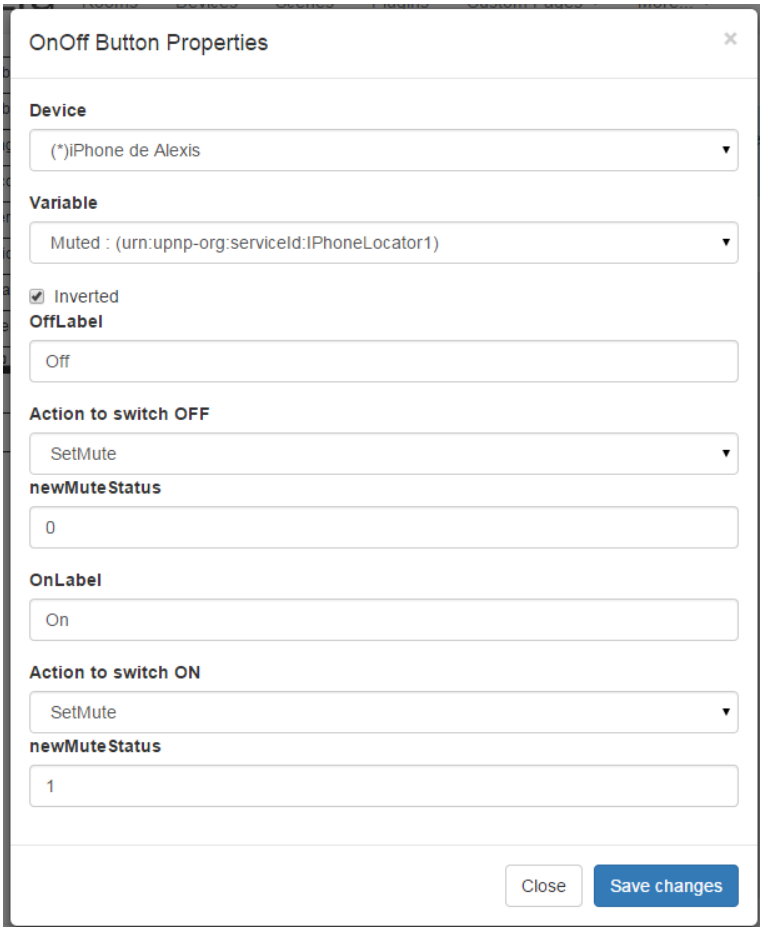
- The lasso (dotted line) rectangle enabling the selection of multiple controls. Ctrl+Click is also supported
- The alignments tools in the left tool bar
- The resize handle at the bottom right corner of the image enabling you to size the image.



This will continue to evolve to add some more new tools (which can be VERA related or even something totally different like a google chart gauge or whatever)

- New 2 state button tool
- New Google gauge with customizable min max & color ranges


Vera™ • Smarter Home C
UI5
Alternate UI to UI7
VERA AltUI
192.168.1.16/port_3480/data_request?id=lr_ALTUI_Handler&command=home#
Vera
Rooms
Devices
Scenes
Plugins
Custom Pages
Mo
Label
Variable
Image
Device Icon
Scene
Action
Multi State
Camera
Gauge
Custom Pages Editor
Drag and Drop to add/move/delete controls. use Ctrl+Click or lasso
Salon
Actions
Off
Off
Alexis
0.022 Km
On
Frede
20.148Km
On
Ext
10
Piscine
8
Caves
Watts
155


OnOff Button Properties
Device
(*)iPhone de Alexis
Variable
Muted : (urn:upnp-org:serviceId:IPhoneLocator1)
Inverted
OffLabel
Off
Action to switch OFF
SetMute
newMuteStatus
0
OnLabel
On
Action to switch ON
SetMute
newMuteStatus
1
Close
Save changes

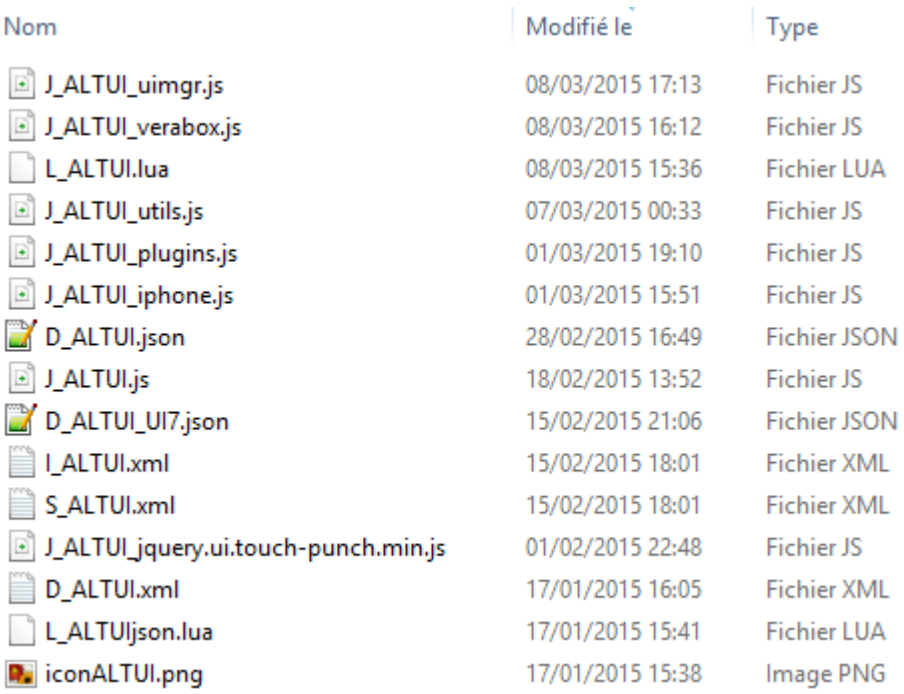
UI5 Installation Instructions (similar for UI7)

PREFERED METHOD:

- Install from store http://apps.mios.com/plugin.php?id=8246
- Then override with latest version where xxx is the latest revision number : http://code.mios.com/trac/mios_alternate_ui/changeset/xxxxx/?old_path=%2F&format=zip

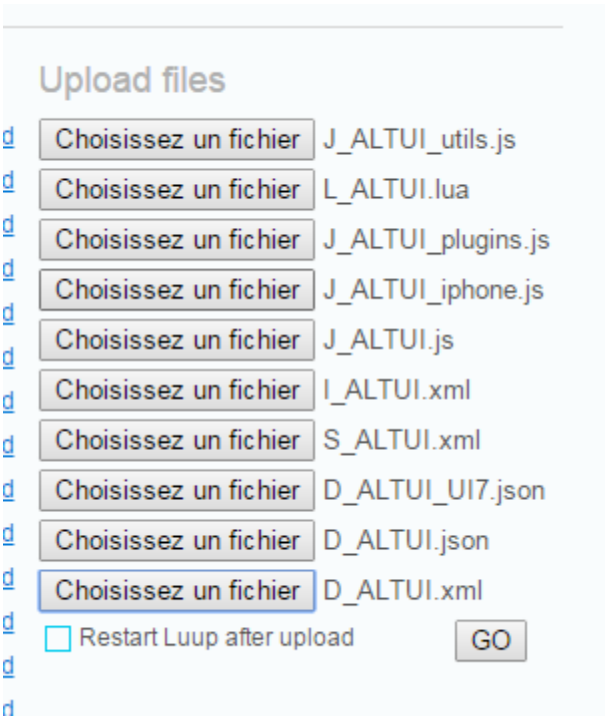
DETAILS

1) Upload all these files



Nom	Modifié le	Type
J_ALTUI_uimgr.js	08/03/2015 17:13	Fichier JS
J_ALTUI_verabox.js	08/03/2015 16:12	Fichier JS
L_ALTUI.lua	08/03/2015 15:36	Fichier LUA
J_ALTUI_utils.js	07/03/2015 00:33	Fichier JS
J_ALTUI_plugins.js	01/03/2015 19:10	Fichier JS
J_ALTUI_iphone.js	01/03/2015 15:51	Fichier JS
D_ALTUI.json	28/02/2015 16:49	Fichier JSON
J_ALTUI.js	18/02/2015 13:52	Fichier JS
D_ALTUI_UI7.json	15/02/2015 21:06	Fichier JSON
I_ALTUI.xml	15/02/2015 18:01	Fichier XML
S_ALTUI.xml	15/02/2015 18:01	Fichier XML
J_ALTUI_jquery.ui.touch-punch.min.js	01/02/2015 22:48	Fichier JS
D_ALTUI.xml	17/01/2015 16:05	Fichier XML
L_ALTUIjson.lua	17/01/2015 15:41	Fichier LUA
iconALTUI.png	17/01/2015 15:38	Image PNG

Example:

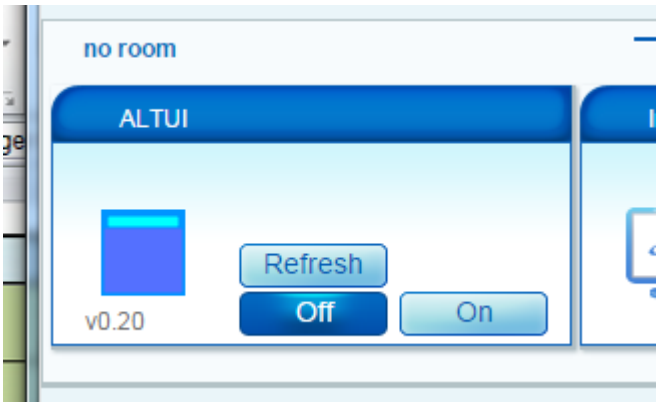


2) Create a device

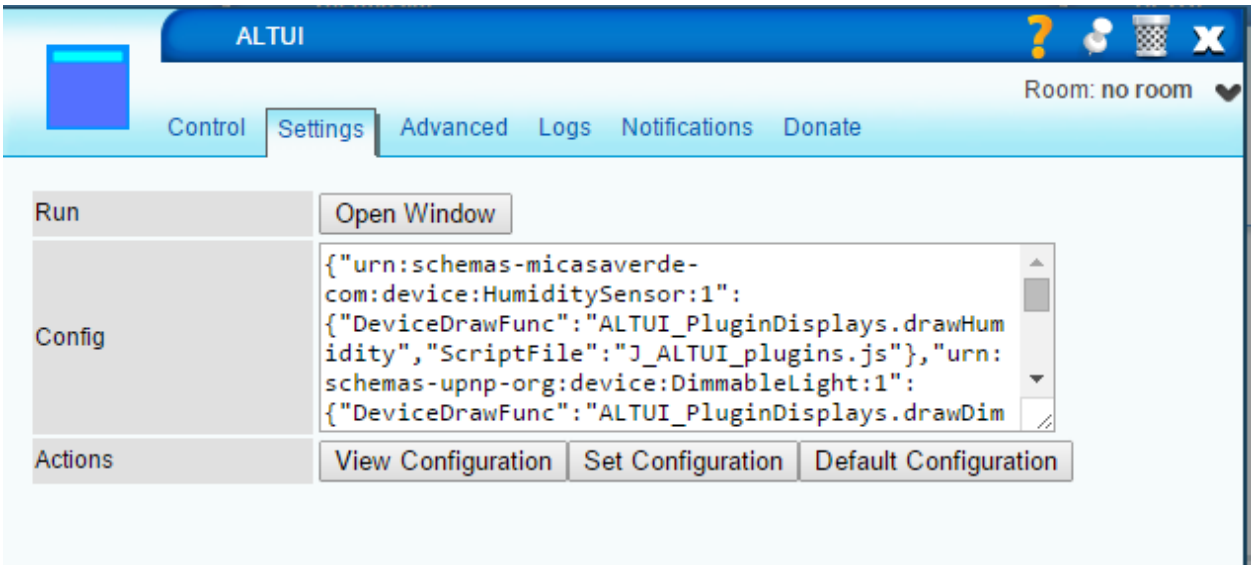
ONLY DO THIS IF THE DEVICE DOES NOT ALREADY EXIST. If you installed from the store, the device has been created automatically for you

3) Reload lua

4) Find the device in UI5

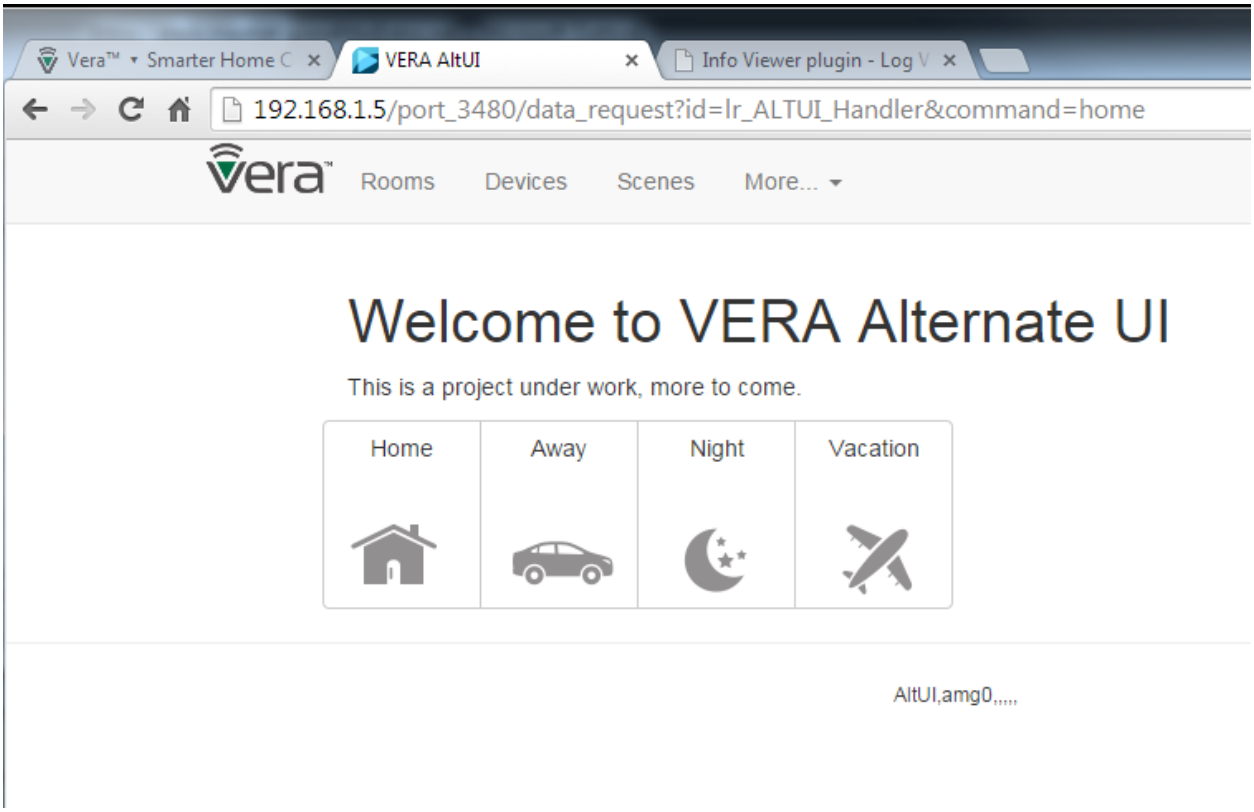


5) Open the settings tab



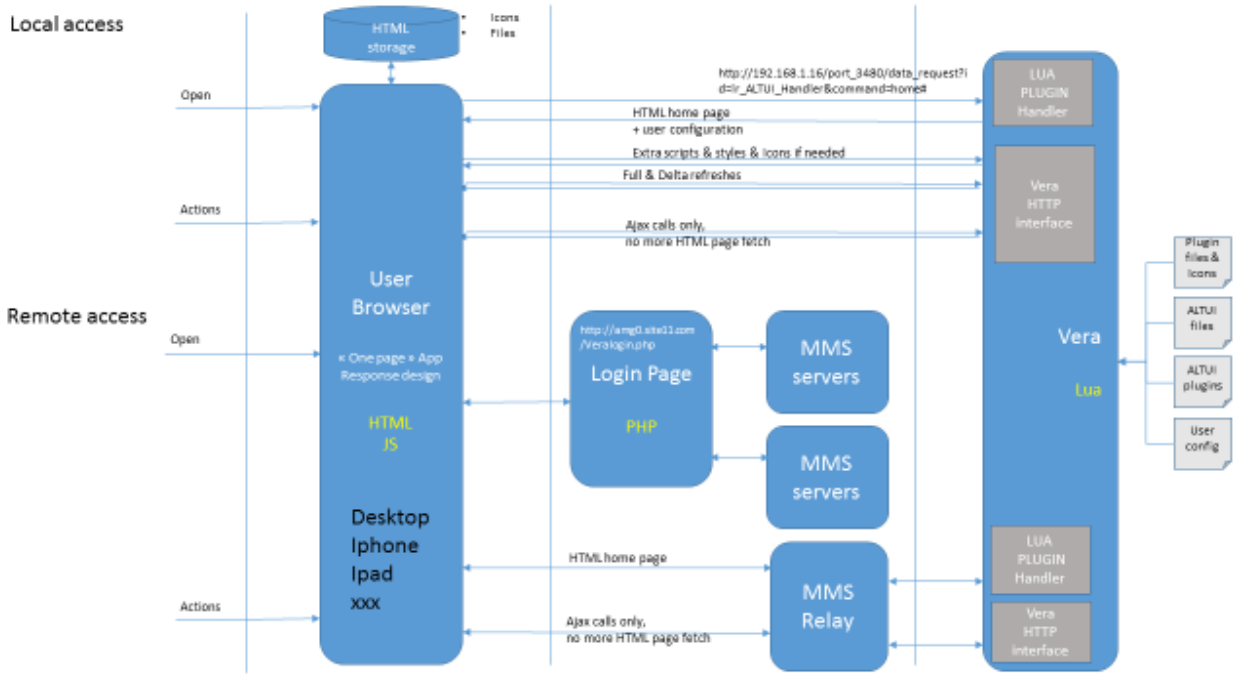
- View configuration : view the JSON configuration object in a JSON online viewer
- Set configuration : set the ALTUI plugin configuration
- Default : reset to default

6) Click on open window
    a. “modes” cannot work on UI5 of course but rest should be ok



Architecture and Source Code organization

Data Flows



Extensibility

Mechanisms to extend

- Device dashboard drawing js function
- Device control panel js function
- New pages can be added (Upnp devices, IP devices, Custom user pages , floor plan dashbnoard , google gauges etc)

Javascript modules for customizable plugins

Plugins drawing are javascript modules providing function code and style css necessary. All the modules are loaded dynamically when needed

Plugins can customize/extend the drawings of device for 2 distinct scenario.

a) the small device box on the Device page
b) a control panel, dedicated for one device, having almost the full page to play with and display specific device status , controls, drawings etc... (I added this "control panel" feature just in the drop down menu under Variable & Actions items.)

A default implementation is provided for both obviously. Right now the “control panel” one is useless and work in progress but I demonstrate the ability on 2 devices uses a custom control panel function. the Binary Light and the IPhone Locator. The point for me was to explain / demonstrate the extensibility

of the architecture and how it would work. if JS developpers / plugin authors have interest to create a control panel for their device (or some other device), we can integrate their work easily in independent modules

Now a bit on the "how":

- each device type can have a custom javascript file. this is declared in the .LUA file L_ALTUI.lua. The “PluginConfig” LUA device variable contains the JSON object for this configuration and **can be modified to add new plugins.**
- in the configuration, for a given device type you can specify a script file (["ScriptFile"]="J_ALTUI_plugins.js",) , a small device box drawing function (["DeviceDrawFunc"]="ALTUI_PluginDisplays.drawBinaryLight",) , a full blown control panel drawing function (["ControlPanelFunc"]="ALTUI_PluginDisplays.drawBinLightControlPanel",) and a style function for your own CSS (["StyleFunc"]="ALTUI_PluginDisplays.getStyle",). All these are optional, default implementation is provided in any case. All these scripts & functions are dynamically loaded and executed when needed by the main page.
- The declared function can be qualified by any number of module name thus enabling to use the javascript module object pattern.
    - Function can be ‘myfunction()’
    - Or ‘myModule.mySubModule.myFunction()’ (any depth)
- see examples of Style and drawing functions in J_ALTUI_plugins.js or J_ALTUI_iphone.js

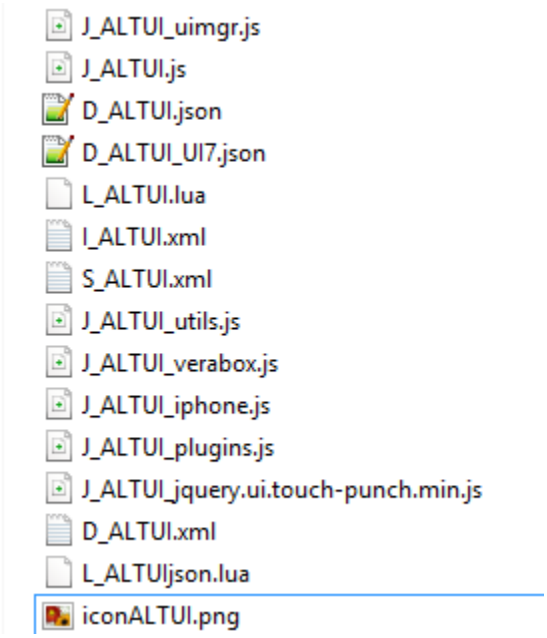
Note:

- the ["DeviceDrawFunc"] takes input parameters like (devid, device) and must return a string which the HTML going into the small device box on the main page.
- the ["ControlPanelFunc"] works slightly different in order to give almost full DOM control to the code writer. It takes input parameters like (devid, device, domparent (jquery based) and must write directly its HTML code into the domparent object (using $.(domparent).append(...)). it felt more comfortable for the contralPanel function to really write in the DOM as they almost own the full page this time.

Full Source code

all code available on http://code.mios.com/trac/mios_ipx800/browser/trunk/AltUI so feel free to try if you are interested.

Source Files:



- **J_ALTUI_uimgr.js**
  - Implements the UIManager object. This object is in charge of all drawing on pages
    - Error messages
    - Device Drawing (default & custom) – manages the loading of JS files needed. Evaluate Icon conditions based on existing UI5 or UI7 descriptions.
    - Scene Drawing (and editor)
    - Refresh UI (when new data is arriving)
    - The main entry points (pagesxxx() function per each page of the app)
    - House mode on UI7 only . the LUA plugin tells the application if we are on UI5 or UI7
  - It maintains internally the cache for device type information (json, Upnp descriptions etc)
- **J_ALTUI.js**
  - The classical JS for the setting page of the UI5 or UI7 plugin
- **D_ALTUIxx.json**
  - The classical JSON files for the UI5 or UI7 plugin
- **L_ALTUI.lua**
  - The main plugin lua code
  - It is seldomly used, just to register a handler to act as a web server serving initially the first home page layout, and eventually responding to a few (one so far) ajax call from the client. The idea is to not use it as much as possible to offload the application work on the client side as explained in the initial project rules. VERA is small, our PC are big
  - It will act as the data persistent place where configuration and (in the future) user custom pages descriptions are stored & saved as device variable. UI7 can store and

display JSON in its variable. UI5 has trouble to display it in the advanced tab as the string are not htmlENcoded but this is not a problem; we can manage this display & edit in the plugin JS setting page.

- **I_ALTUI.xml , S_ALTUI.xml**
  - Classical device files.
  - A Reset uPNP action is implemented to restore configuration to default
  - In the future, we may need a few to manage user custom pages, not sure yet
- **J_ALTUI_utils.js**
  - Global utilities like string.format() addition, ro string.htmlEncode(), htmlDecode() addition to the string prototype
  - CSS Styles required by the application are managed here and injected dynamically (avoid having to change the .LUA file and reloading every time)
  - It initializes the application by launching the Init() for the UIManager object and the VeraBox object
- **J_ALTUI_verabox.js**
  - Implements the communication with VERA
  - The UPnpHelper module
    - facilitates building of URL (get set variables, run upnp, all the VERA Http calls basically including the HAG SOAP one)
    - Provides facitilies for plugin author like simple SetOnOff() , SetArm() methods
  - The FileDB module
    - A cache of dynamically loaded files (D_xx files S_xx files, or whatever). Key is the file name.
    - In the future, I intent to use HTML5 persistent storage to cache content on a even longer term basis (even when user closes the browser)
  - The VeraBox module
    - The core data load engine.
    - Manages getting the user_data and status_data using the loadversion versioning and various optimizations documented
    - Manages all information in a cache to not load it twice
    - Highly asynchronous, code executed in callback methods instead of waiting
      - getWeatherSettings : _getWeatherSettings,
      - getBoxInfo : _getBoxInfo,
      - getLuaStartup : _getLuaStartup,
      - getRooms : _getRooms, // in the future getRooms could cache the information and only call _getRooms when needed
      - getDevices : _getDevices,
      - getDeviceByID : _getDeviceByID,
      - getScenes : _getScenes,
      - getSceneByID : _getSceneByID,

- getPlugins : _getPlugins,
- getHouseMode : _getHouseMode,
- setHouseMode : _setHouseMode,
- getStatus : _getStatus,
- getStates : _getStates,
- evaluateConditions : _evaluateConditions, // evaluate a device condition table (AND between conditions)
- deleteRoom : _deleteRoom,
- runScene : _runScene,
- deleteScene : _deleteScene,
- reloadEngine : _reloadEngine,
- setStartupCode : _setStartupCode,
- setScene : _setScene,
- getCategoryTitle : _getCategoryTitle,
- getDeviceTypes
- initEngine()

- **J_ALTUI_IPhone.js**
  - The custom drawing functions for the IPHone locator plugin and the French Canal Plus control Plugin
  - Dynamically loaded when/if needed and configured in the LUA "PluginConfig" table to be loaded
- **J_ALTUI_Plugins.js**
  - Same but for all the out of the box devices provided by VERA (bin lights, motion, temp sensors, etc)
- **J_ALTUI_jquery.ui.touch-punch.min.js**
  - A small jquery 3rd party to make the ipad/iphone/ touch screen device compatible with the click event () so that touchend event can be used as a mouse click

Basic rules for developers:

I most welcome any programmers help in this project if they are interested in submissions. The rules are simple,

- use bootstrap grid model (row / cols) for full responsive design, I d like to keep it running from desktop to ipad to iPhone 4S screen !
- minimize additional JS framework : I am trying to use bootstrap, jquery, jqueryUI and that's all. probably would liek a try to add a lib for graphic widgets (gauges etc) also in the near future, suggestion welcome. must be easy and working with jquery as I am far from a JS guru

- use JS module pattern (same as UI7) see example in the various modules. prefix private function with a '_' and public function with a naming convention doSomethingToSomethingElse()
- all CSS class: try to always use the prefix : altui-xxx-xxx etc
- avoid synchronous call when possible (always possible almost)