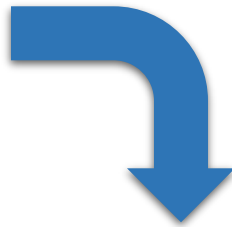
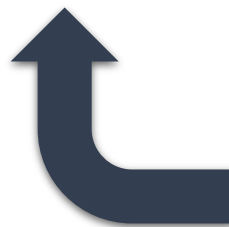


openHAB Application for IP POWER (For Windows Version)

Manual updated: 2024/06 Ver.1.1



openHAB
empowering the smart home



Contents

| | |
|---|----|
| 1.openHAB installation..... | 4 |
| 1.1 Install the Bindings..... | 10 |
| 1.2 Preparation for IP Power Configuration..... | 12 |
| 2.Setting IP Power for 9255, 9258 series..... | 13 |
| 2.1 Create things..... | 13 |
| 2.2 Create Channel | 17 |
| 2.3 Create Item | 21 |
| 2.4 Configuration Parameters Corresponding to the 9258 Series Code..... | 25 |
| 3. Setting IP Power for 9850, 9858MT, 9855 series..... | 26 |
| 3.1 Create things..... | 26 |
| 3.2 Create Channels | 28 |
| 3.3 Create item..... | 30 |
| 3.4 Configuration Parameters Corresponding to the 98 Series Code | 30 |
| 3.5 Create attributes Channel..... | 31 |
| 4. MQTT setting | 36 |
| 4.1 Create MQTT Broker | 37 |
| 4.2 Create Generic MQTT Thing..... | 38 |
| 4.3 Create Generic MQTT Thing Channels..... | 39 |
| 4.4 Create items..... | 42 |
| 4.5 MQTT Code | 42 |
| 5. openHAB App LAN and WAN settings | 43 |
| 6. IP Power's code in openHAB | 51 |
| 6.1 9258 code..... | 51 |
| 6.2 98 series code | 52 |
| 6.3 Generic MQTT thing code | 54 |
| 6.4 Attribute Settings: (e.g., Current, Voltage, Temperature) | 56 |

AVIOSYS's IP POWER can now be used in Home Assistant. In this system, you can set your IP POWER according to your own needs. Please follow the steps in the text:

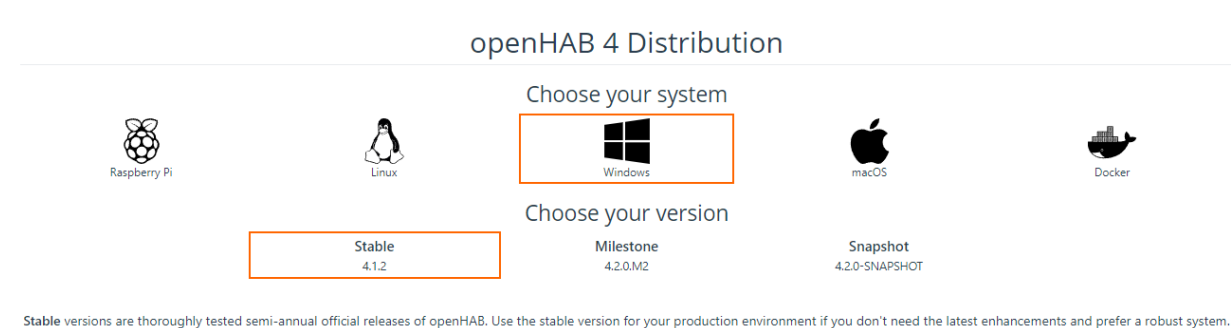
1. openHAB installation

Please move to the official openHAB download page and select the Windows version. Follow the instructions to start the installation.

Official link: <https://www.openhab.org/download/>

When you select the OS you want to install, three versions of openHAB will appear below.

This tutorial focuses on Stable Version 4.1.2.



Please follow the official instructions to download the JAVA 17 platform, and then download Azul Zulu Builds of OpenJDK first.

You can click on the links provided by the official source to download.

Manual Installation

1. Install a recent Java 17 platform (we recommend **the Zulu builds of OpenJDK**)

***Note:** This installation is a prerequisite. Without installing this platform, openHAB cannot be activated.

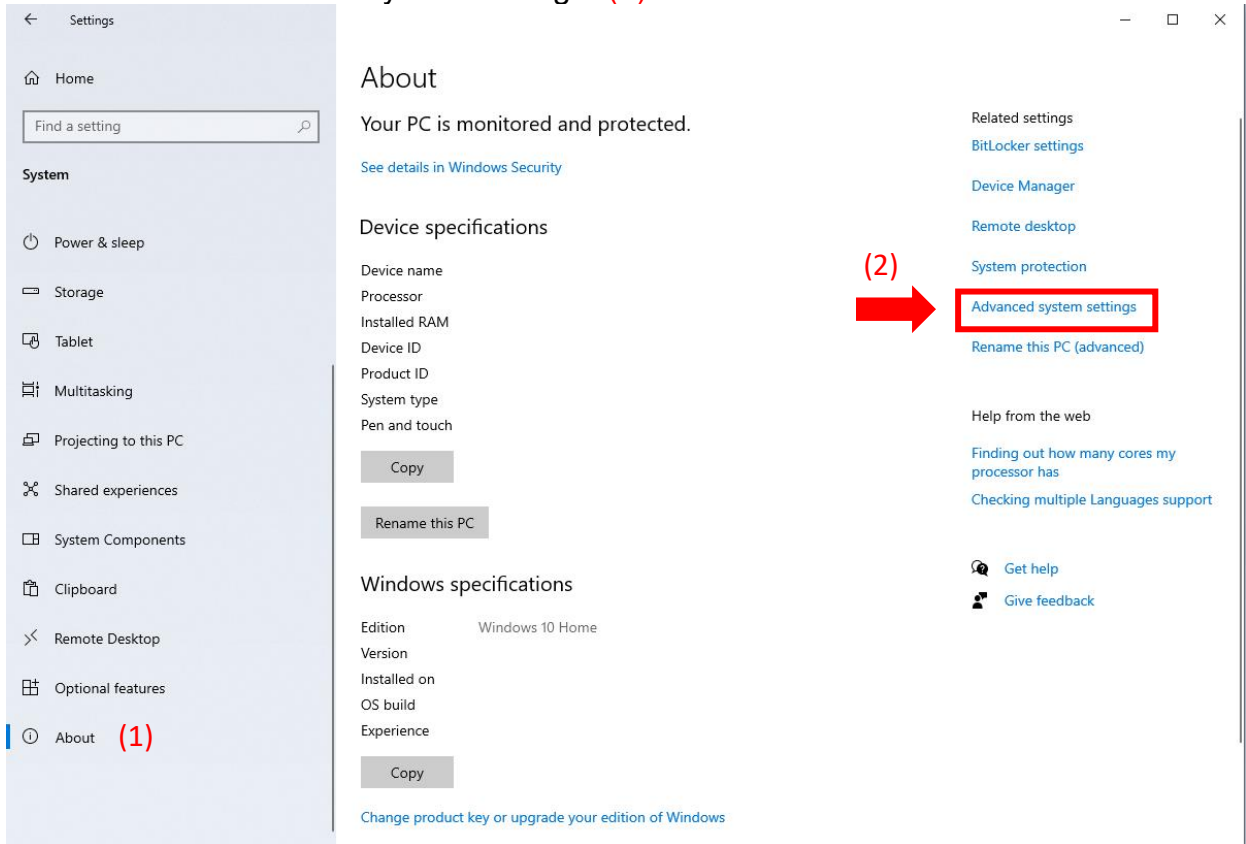
After downloading and completing the installation, please follow the instructions on the next page to complete the Azul setup.

Azul setup:

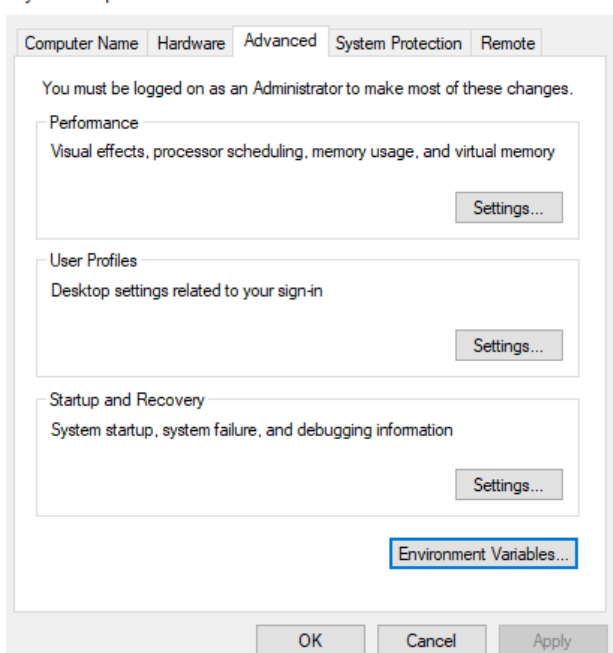
Click on the Windows icon on the desktop , then click on the settings icon , then select the system.

After entering the system window, click on "About"(1) on the left to enter the About page.

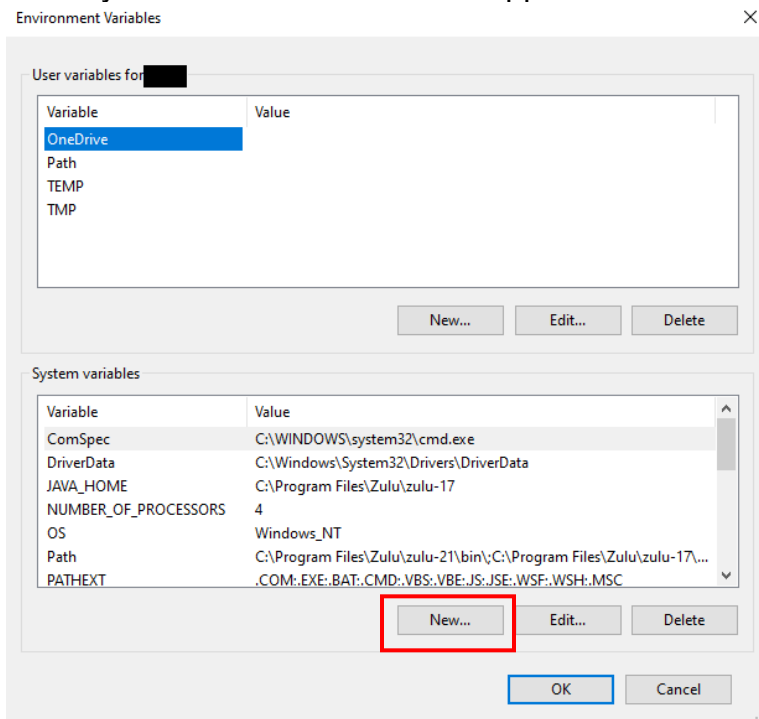
Then click on "Advanced system settings."(2)



After entering Advanced system settings, please click on Environment Variables.



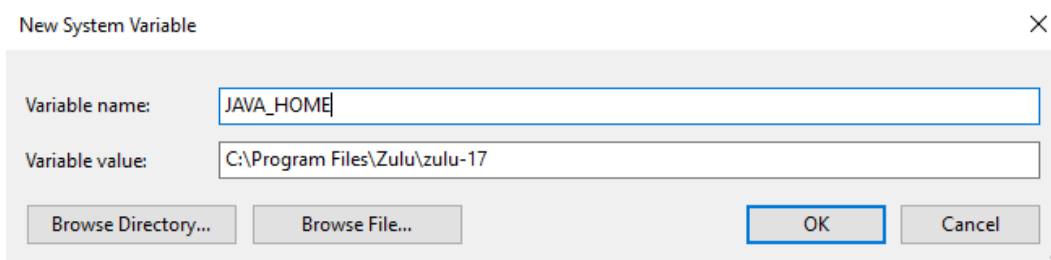
Once you enter Environment Variables, click on the new button below. Upon clicking, a New System Variable window will appear.



After the appearance of the New System Variable, please fill in the following information in Variable name and Value.

- (1) Variable name: JAVA_HOME
- (2) Variable value: Enter the installation location of Azul Zulu.

After you finished, click OK below to complete the installation of Azul Zulu.



openHAB installation:

Returning to the openHAB installation page, proceed to the next step.

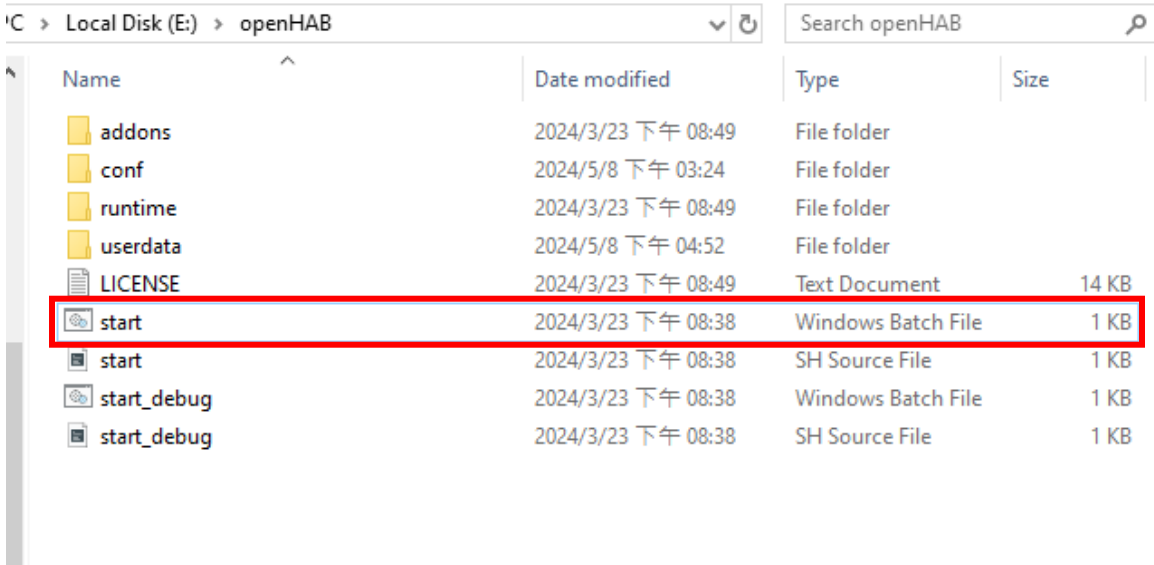
- (1) Click on the "Download openHAB 4.1.2 Stable Runtime" in the download section. Once downloaded, begin the installation.

After installation is complete, please create a folder for openHAB first, and choose the disk slot to place.

After entering the openHAB folder, please unzip the downloaded openHAB zip file.

Once the extraction is complete, it will appear as shown in the image below. After confirming the extraction,

click on "start" (Windows Batch File) inside the folder.



When the program starts, it takes about 10 to 30 seconds to establish a connection with openHAB.

***Note:** If this screen persists for an extended period, it indicates that the program is not running properly. Please check the previous steps or restart 'start'.



After successful operation, the openHAB icon will appear. Please do not close this program while using openHAB.



```
Karaf
Launching the openHAB runtime...

  openHAB
    4.1.2 - Release Build

Use '<tab>' for a list of available commands
and '[cmd] --help' for help on a specific command.
To exit, use '<ctrl-d>' or 'logout'.

openhabs>
```

Please open any web browser (e.g., Edge, Chrome) and enter: <http://localhost:8080>
Once on the webpage, create your openHAB account.



Create a first administrator account to continue.


User Name

Password

Confirm New Password

Create Account

After creating your account, you will enter the configuration page. You do not need to make initial configurations immediately; you can skip them and go directly to the main page.



Language 英文 >

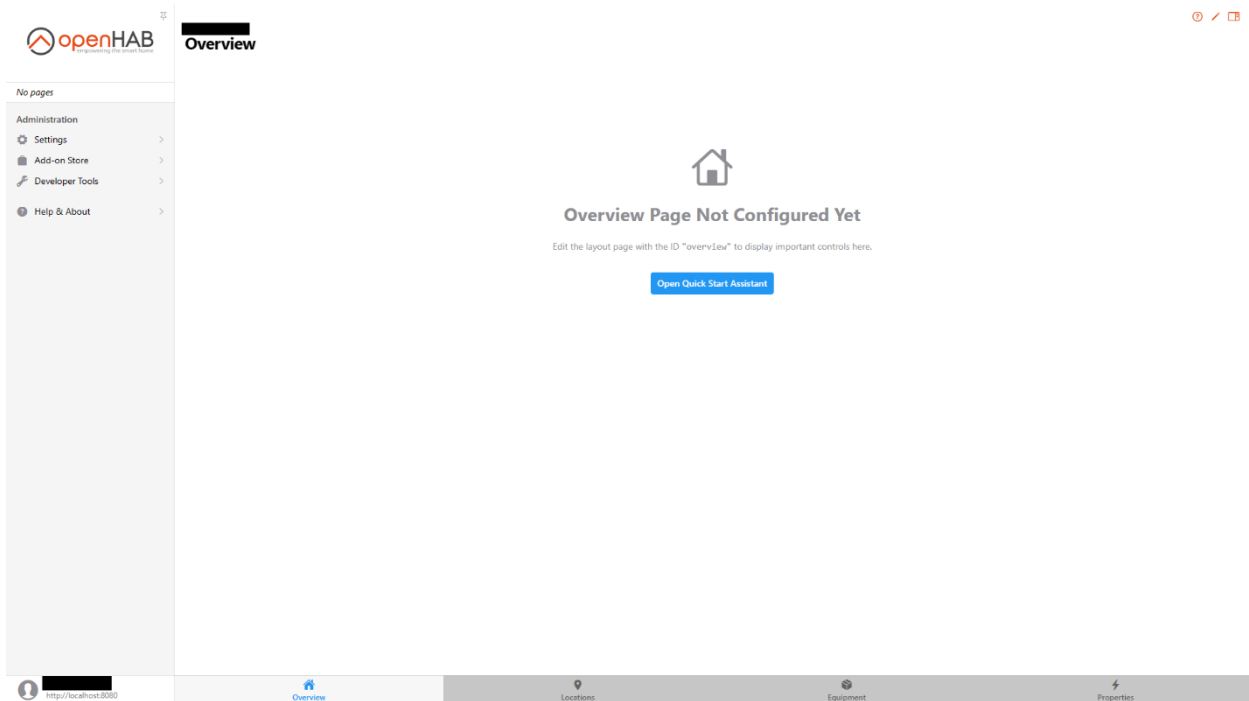
Region 美国 >

Time Zone (GMT-8:00) America/Los_Angeles >

[Begin Setup](#)

[Skip Setup](#)

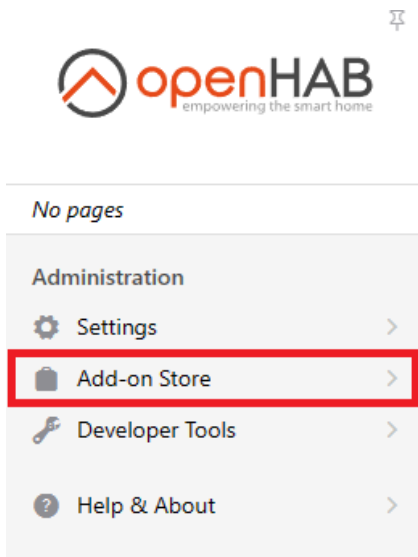
When you enter this screen, it means you have successfully set up your openHAB.



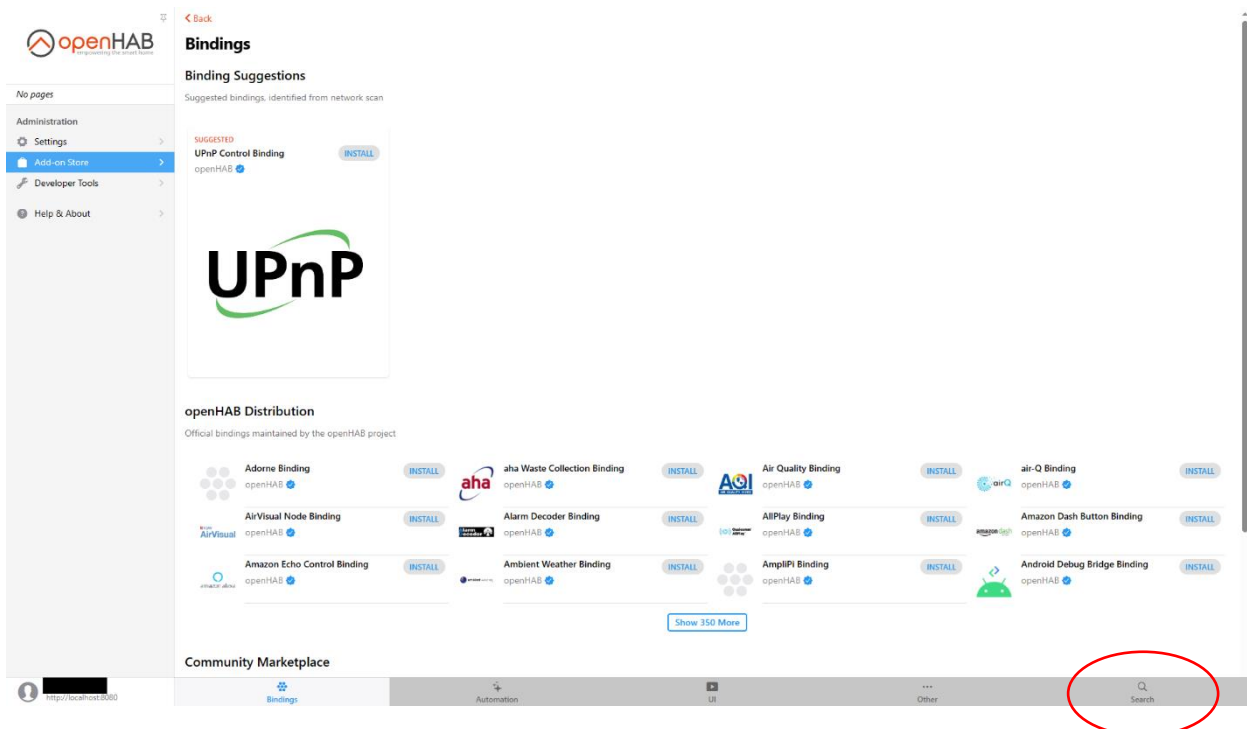
The screenshot shows the openHAB Overview page. On the left is a sidebar with the openHAB logo and a menu: Administration (Settings, Add-on Store, Developer Tools), and Help & About. The main content area has a header with the openHAB logo and the title 'Overview'. Below this is a large house icon and the text 'Overview Page Not Configured Yet'. A subtext says 'Edit the layout page with the ID "overview" to display important controls here.' and there is a blue button 'Open Quick Start Assistant'. At the bottom is a navigation bar with icons for Overview, Locations, Equipment, and Properties. The user's profile and URL are visible in the bottom left corner.

1.1 Install the Bindings

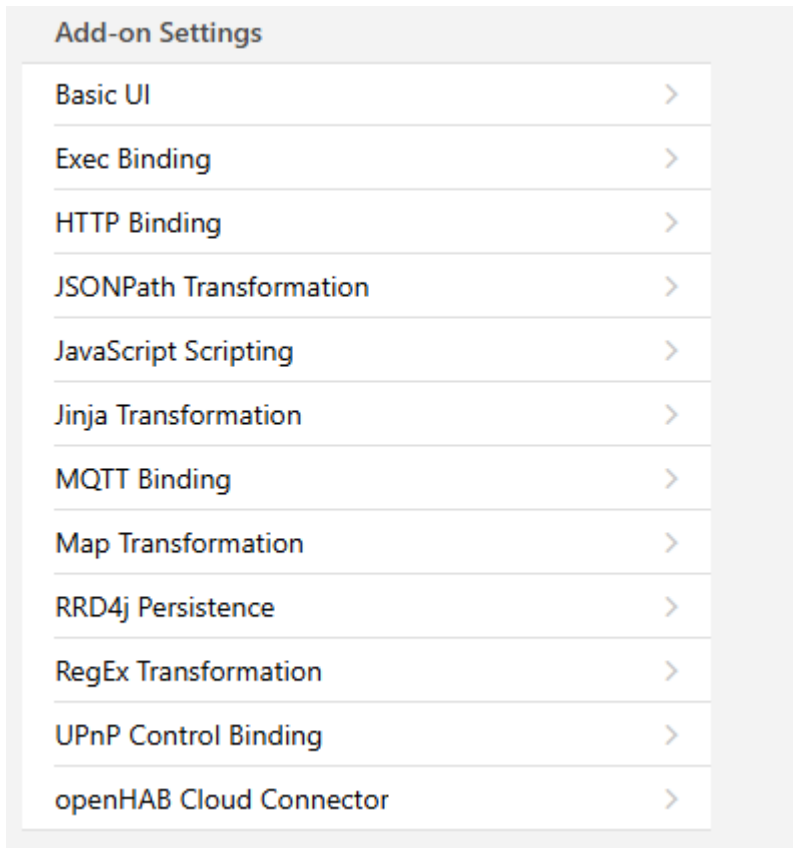
Before setting up IP Power with openHAB, you need to install some “Bindings” first. Please click on "Add-on Store" on the left side of the main page.



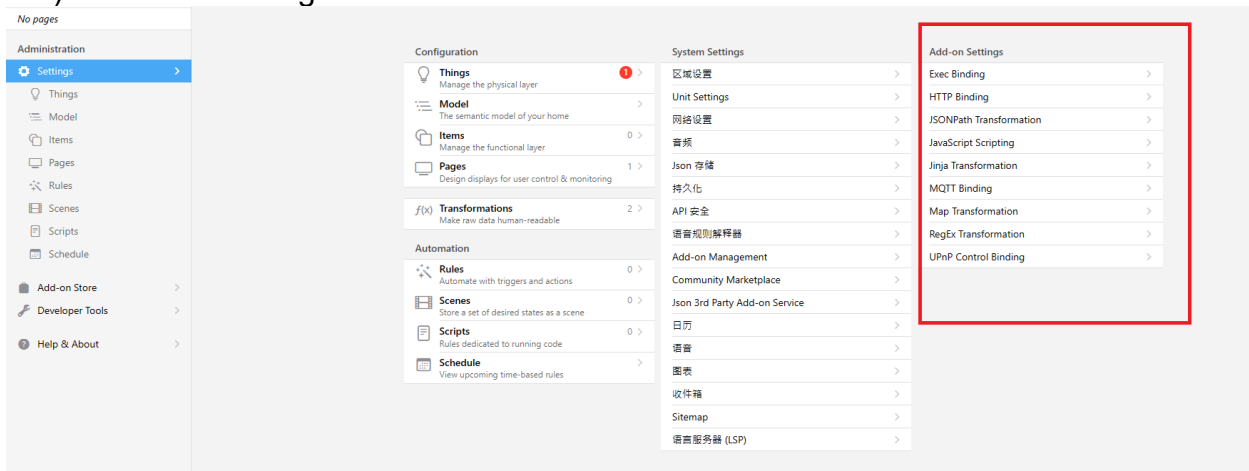
Once on the page, please click on "Search" at the bottom right corner.



On the Search page, please proceed to install the bindings in the Add-on Settings below.

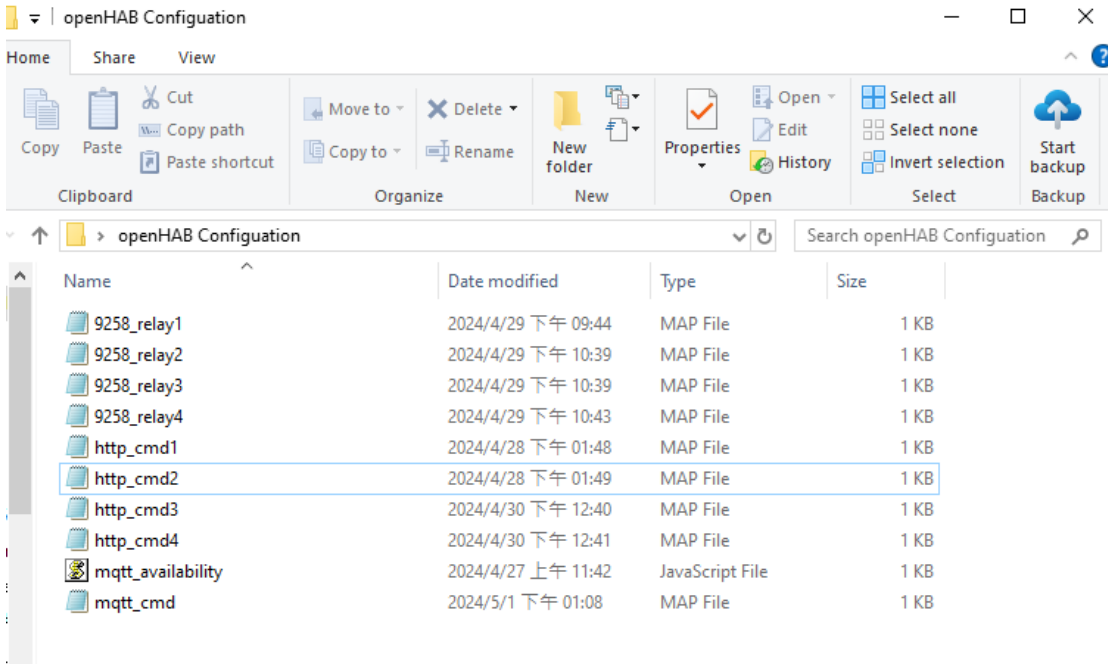


After installation, all bindings will appear in the "Add-on Settings" section (highlighted in red) within the Settings interface.

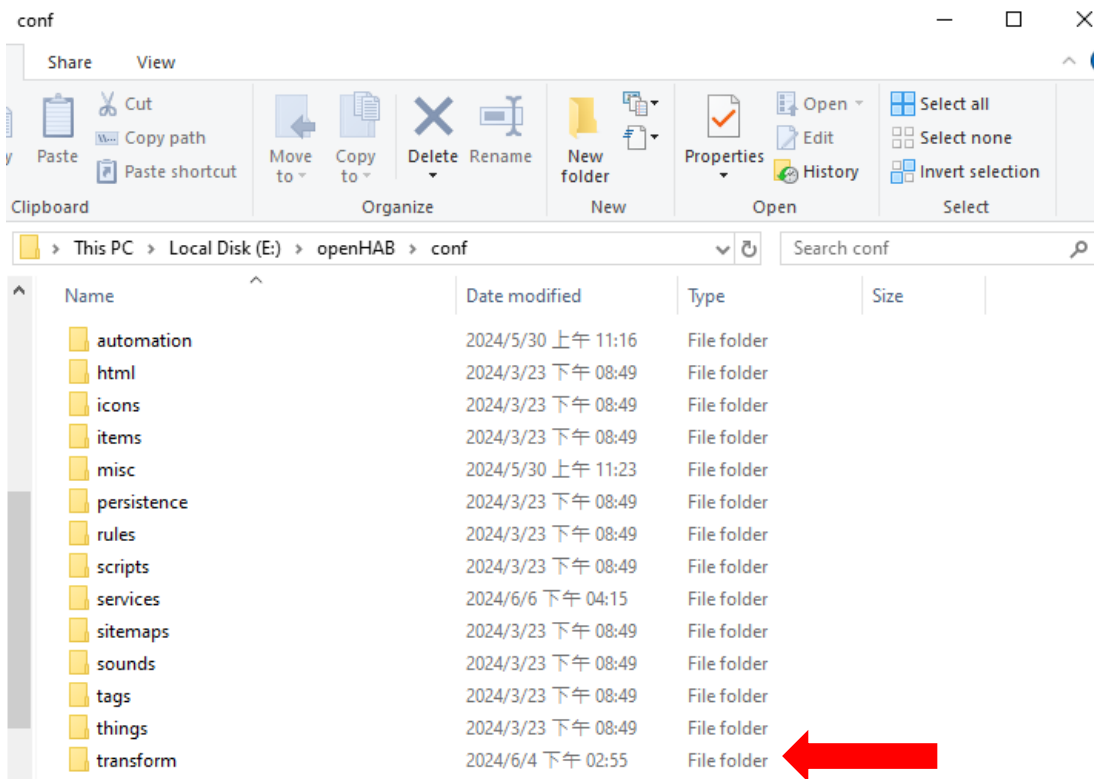


1.2 Preparation for IP Power Configuration

Please download the manual from the AVIOSYS official website, click on "openHAB configuration," and place these files initially in openHAB/conf/transform. Failure to do so will prevent the completion of subsequent device setups.



The location: e.g.: E:/openHAB/conf/transform

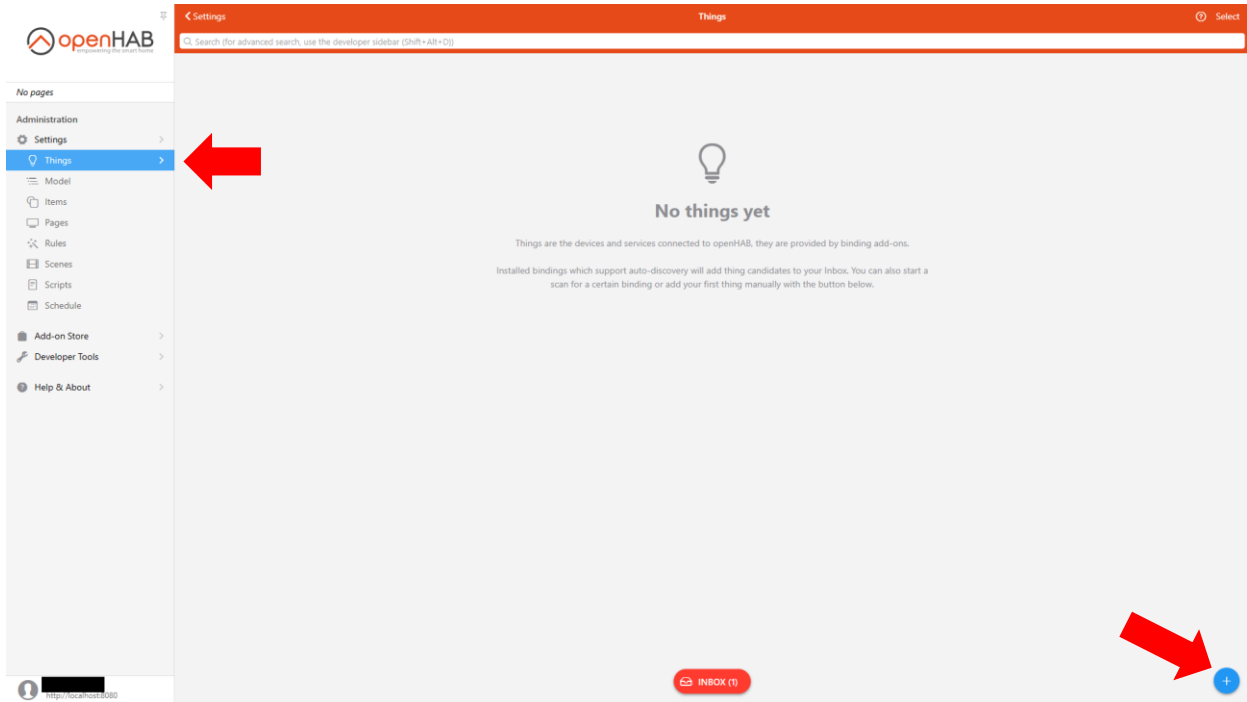


2. Setting IP Power for 9255, 9258 series

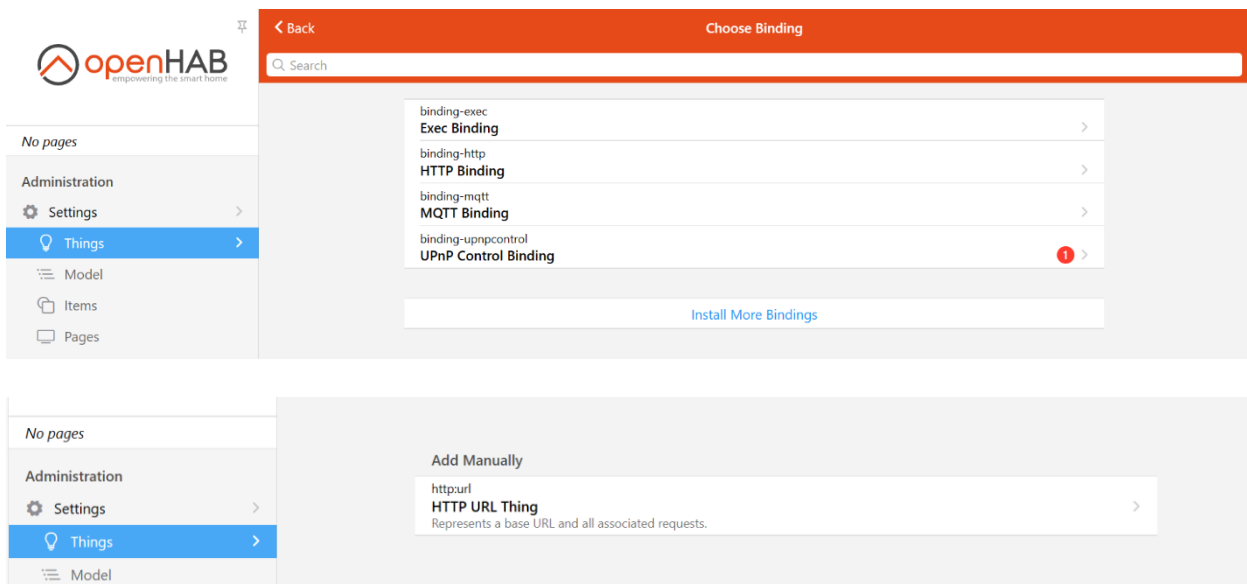
***Note:** Some steps are related to section 1.2, please refer to P.12 and prepare in advance.

2.1 Create things

Please click on Settings on the left, then select Things. After entering the Things screen, click the blue icon at the bottom right.



Once on the screen, select HTTP Binding, then click on HTTP URL Thing.



Enter the settings page and fill in the required configuration details for IP Power, following the instructions in the image below:

Configuration Steps:

Example model: IP Power 9258

(1) After entering HTTP URL Thing, click Show advanced.

New HTTP URL Thing

| | |
|------------|---|
| Unique ID | e9dbd477bb <small>Note: cannot be changed after the creation</small> |
| Identifier | http://url:e9dbd477bb |
| Label | HTTP URL Thing |
| Location | e.g. Kitchen |

HTTP URL Thing
Represents a base URL and all associated requests.

Show advanced ☐

Base URL
Required The URL set here can be extended in the channel configuration.

Refresh Time
30
Time between two refreshes of all channels

Timeout
3000
The timeout in ms for each request

Create Thing

(2) Please fill in the configuration information required for IP9258 according to the icons in the image below. The explanation of the icons is as follows:

- ✓: Please directly copy the content from the image into your openHAB
- ✗: Do not directly copy and paste; some settings may differ depending on your location. (e.g., IP address)
- ▲: This is a reminder: if you haven't changed any settings on the device, you can directly copy the content from the image. (e.g., account and password)

HTTP URL setting:

***Notice:** If you want to maintain a long-term stable connection to IP Power, it is recommended to change the connection to "Static" on the IP Power device to prevent the device from automatically changing its IP address.

New HTTP URL Thing

Unique ID

aa77f71ce4

Note: cannot be changed after the creation

Identifier

http:url:aa77f71ce4

Label

IP9258 relay

Location

e.g. Kitchen

HTTP URL Thing

Represents a base URL and all associated requests.

Show advanced

Base URL

http://10.33.122.48

*Notice

Required The URL set here can be extended in the channel configuration.

Refresh Time

30

Time between two refreshes of all channels

Timeout

3000

The timeout in ms for each request

Delay

0

Delay between to requests

Buffer Size

2048

Size of the response buffer (default 2048 kB)

Username

admin

Basic Authentication username

Password

Basic Authentication password

Authentication Mode

Basic Authentication

Preemptive Basic Authentication

Digest Authentication

New HTTP URL Thing

Authentication Mode
☐ Basic Authentication
☒ Preemptive Basic Authentication
☐ Digest Authentication

State Method
☒ GET
☐ POST
☐ PUT
HTTP method (GET,POST, PUT) for retrieving a status.

Command Method
☒ GET
☐ POST
☐ PUT
HTTP method (GET,POST, PUT) for sending commands.

Content Type
☐ application/json
☐ application/xml
☐ text/html
☒ text/plain
☐ text/xml
The MIME content type. Only used for 'POST' and 'PUT'.

Fallback Encoding
Fallback Encoding text received by this thing's channels.

Headers
Additional headers send along with the request

Ignore SSL Errors ☐
If set to true ignores invalid SSL certificate errors. This is potentially dangerous.

Create Thing

After completion, the 9258 Things will be displayed in alphabetical.
 Please click on the IP9258 relay below to complete the remaining settings.

Things

D))

1 Things

Alphabetical

By binding

By location

I
ONLINE
>

IP9258 relay
<http://url:aa77f71ce4>

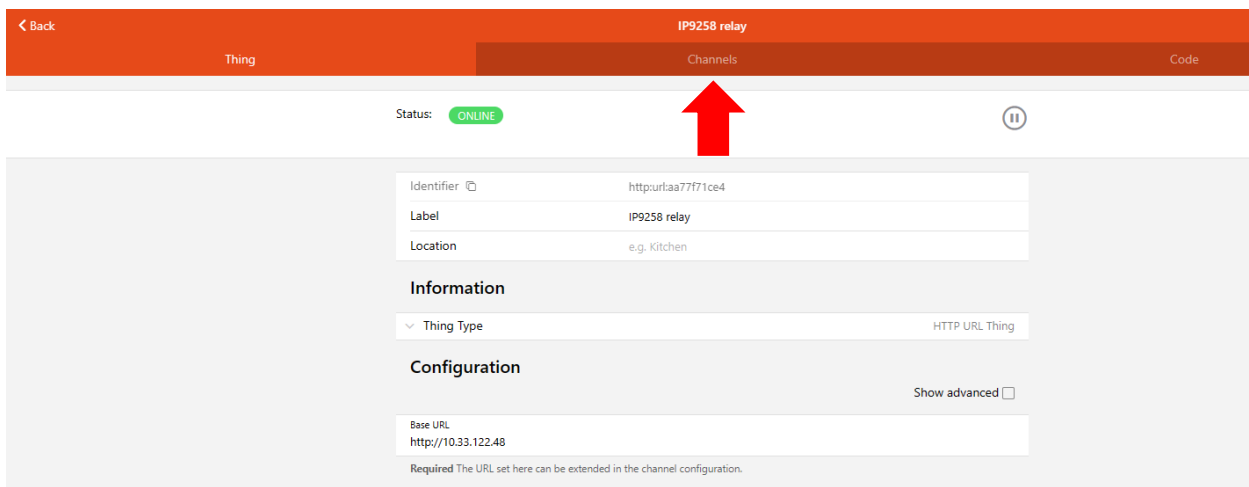
2.2 Create Channel

The 9258 has 4 relay switches, so we need to create 4 Channels. To facilitate the configuration, we have prepared 4 map files. (9258_relay1.map~ 9258_relay4.map) As a mapping for the switches of the commands, these files should be saved under the “ \conf\transform” ° E.g. : D:\openhab-4.1.2\conf\transform °

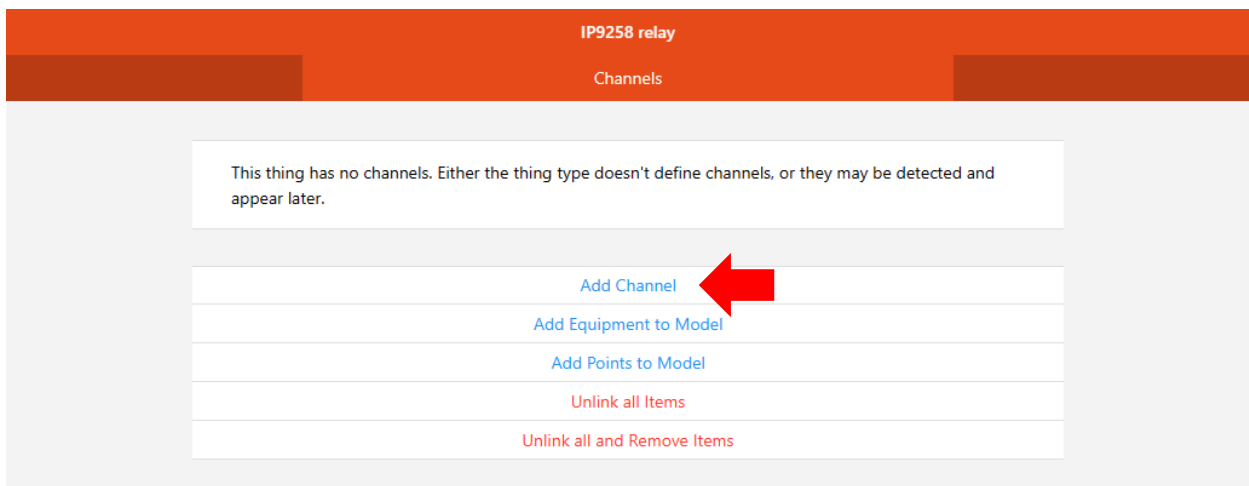
***Note:** To configure the 9255 series, you only need to rename 9258_relay1.map to 9255_relay1.map. Alternatively, you can copy the file, rename it, and place it in the transform folder.

These files are located in the openHAB Channel folder. Please place the .map file for the specified model in the openHAB folder.

When you click on the IP9258 relay Thing, select the Channel indicated by the red arrow in the image."



Then click the Add Channel.



After entering "Add Channel," please follow the instructions shown in the image below:

Note 1: You can change the name according to your preference, but it is recommended to follow the instructions in the image for the initial setup.

Add Channel
IP9258 relay

Note 1

Channel

| | |
|--------------------|---|
| Channel Identifier | id_IP9258_relay1 <small>Note: cannot be changed after the creation</small> |
| Label | IP9258 relay1 |
| Description | |

Channel type

☐ Color Channel

☐ Contact Channel

☐ DateTime Channel

☐ Dimmer Channel

☐ Image Channel

☐ Location Channel

☐ Number Channel

☐ Player Channel

☐ Rollershutter Channel

☐ String Channel

☒ Switch Channel

Add Channel
IP9258 relay

☒ Switch Channel

Configuration

Show advanced

State Transformation

REGEX:^(.*)p61=(\d+).*

Transformation pattern used when receiving values. Chain multiple transformations with the mathematical intersection character "&".

Command Transformation

MAP:9258_relay1.map

Transformation pattern used when sending values. Chain multiple transformations with the mathematical intersection character "&".

State URL Extension

set.cmd?cmd=getpower

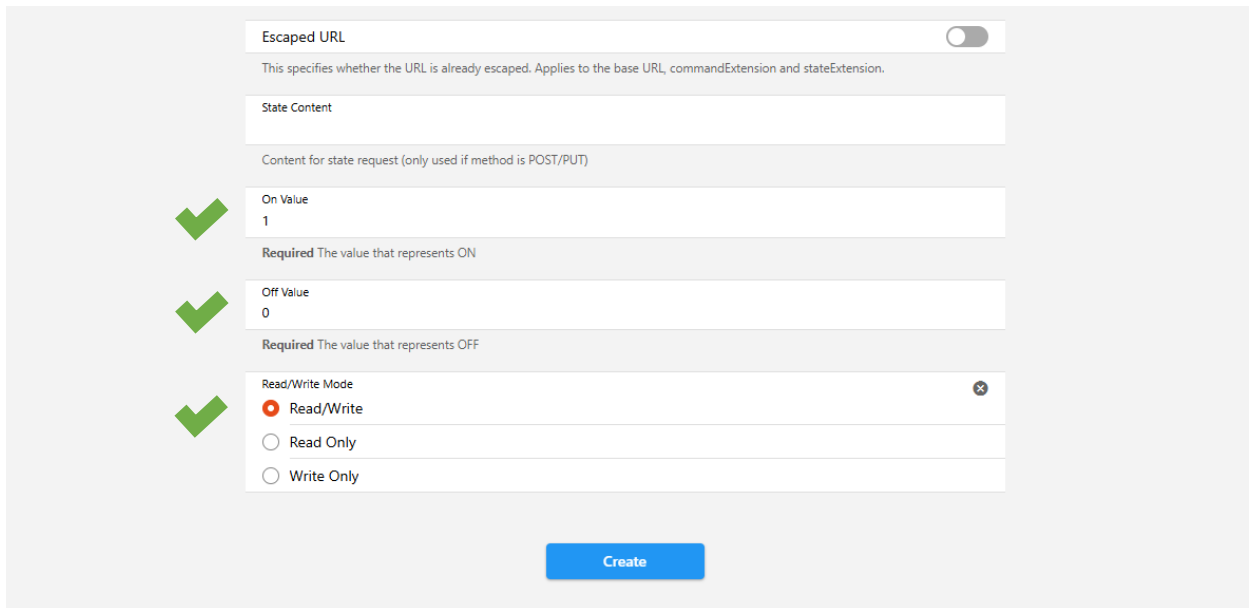
This value is added to the base URL configured in the thing for retrieving values.

Command URL Extension

set.cmd?cmd=setpower+%2\$s

This value is added to the base URL configured in the thing for sending values.

After completing all the settings, please click Create below.

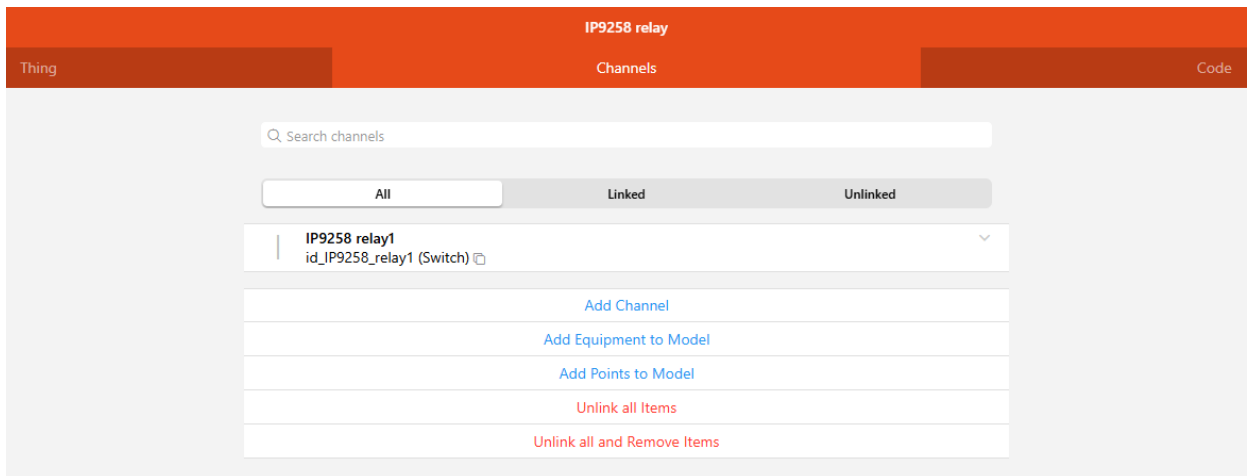


The screenshot shows a configuration form for creating a channel. It includes several sections with green checkmarks on the left indicating successful completion:

- Escaped URL:** A toggle switch is turned on. Below it, a note states: "This specifies whether the URL is already escaped. Applies to the base URL, commandExtension and stateExtension."
- State Content:** A text input field is empty. Below it, a note states: "Content for state request (only used if method is POST/PUT)".
- On Value:** The input field contains the value "1". Below it, a note states: "Required The value that represents ON".
- Off Value:** The input field contains the value "0". Below it, a note states: "Required The value that represents OFF".
- Read/Write Mode:** Three radio buttons are present: "Read/Write" (selected), "Read Only", and "Write Only".

At the bottom center of the form is a blue button labeled "Create".

Once it's complete, your first Channel is finished as the image shown below,



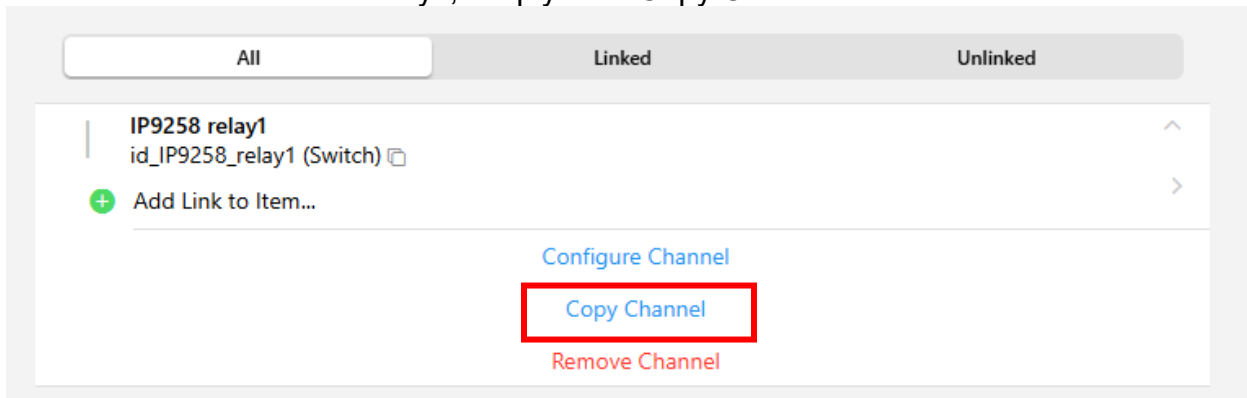
The screenshot shows the "Channels" tab for a device named "IP9258 relay". The interface includes a search bar labeled "Search channels" and three filter buttons: "All", "Linked", and "Unlinked". The "All" button is currently selected.

Below the filters, a dropdown menu shows the selected channel: "IP9258 relay1" with the identifier "id_IP9258_relay1 (Switch)".

At the bottom, there are five action buttons:

- Add Channel
- Add Equipment to Model
- Add Points to Model
- Unlink all Items
- Unlink all and Remove Items

To create the other three relays, simply click Copy Channel below.



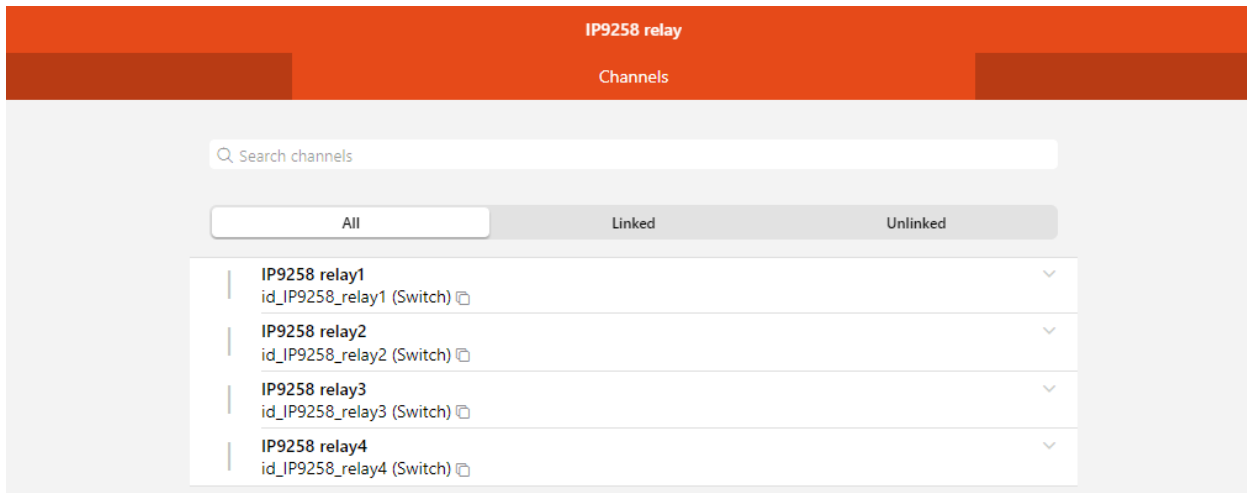
As an example of creating a second relay:

Click "Copy Channel," then change the code name in the red box below the image. No other settings need to be changed. After making the change, please click "Save."

(1) Channel Identifier: `id_IP9258_relay1_copy` → `id_IP9258_relay2`
Label: `IP9258 relay1 copy` → `IP9258 relay2`

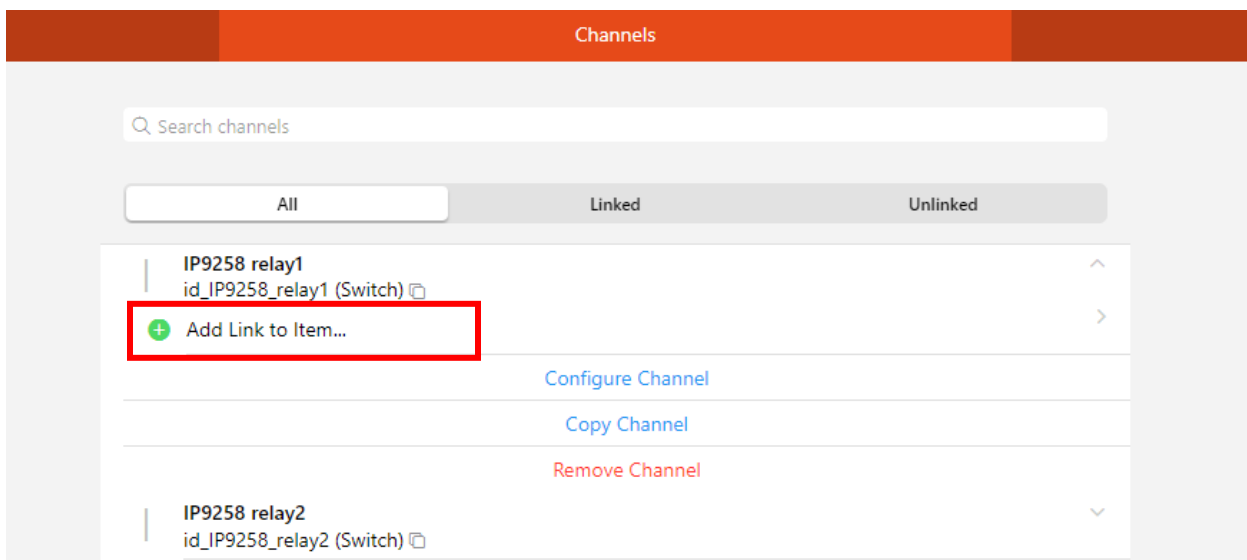
(2) State Transformation: `REGEX:^\.*p61=(\d+).*` → `REGEX:^\.*p62=(\d+).*`
Command Transformation: `MAP:9258_relay1.map` → `MAP:9258_relay2.map`

After completing the Channels for 9258, it will display as shown in the image:



2.3 Create Item

Return to the channel of IP9258, we need to create 4 items. Taking IP9258 relay 1 as an example, please click on "Add Link to Item..." below.



Please configure the settings to match the image below:

Link Channel to Item

Channel

IP9258 relay1
http:url:aa77f71ce4:id_IP9258_relay1 (Switch)

Item

☐ Use an existing Item

☒ Create a new Item

✓

Name

IP9258_relay1

Note: cannot be changed after the creation

✕

✓

Label

IP9258 relay1

✕

Type

Switch >

Category

temperature, firstfloor...

Semantic Class

Point >

Semantic Property

None >

Non-Semantic Tags

▼

After completing the settings, click "Link" to finish the configuration.

Profile

Profiles define how Channels and Items work together. Install transformation add-ons to get additional profiles.
[Learn more about profiles.](#)

✓

☒ 默认

☐ JINJA

☐ JSONPATH

☐ MAP

☐ REGEX

☐ SCRIPT ECMAScript (ECMAScript 262 Edition 11)

☐ SCRIPT Rule DSL (v1)

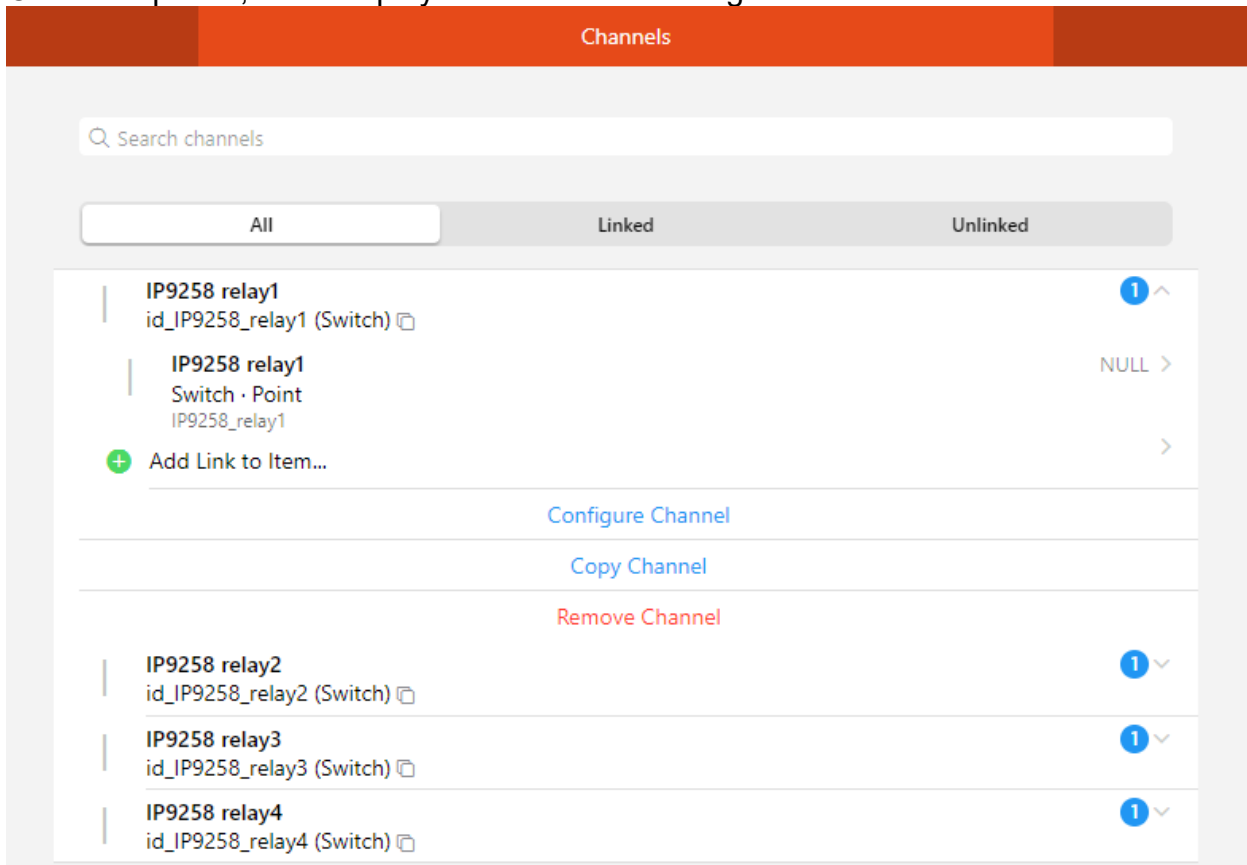
☐ 关注

☐ 更改时的时间戳

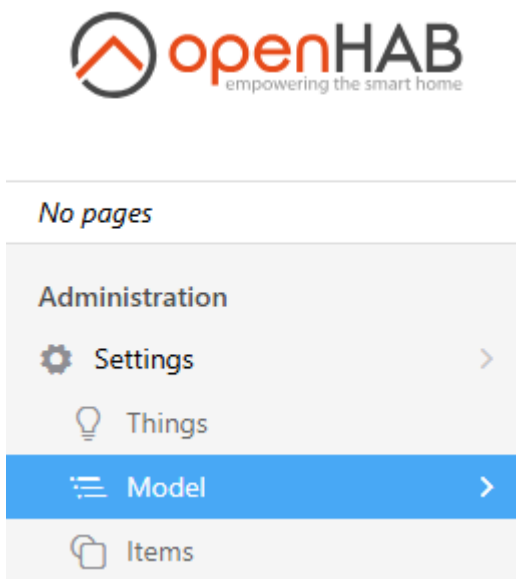
☐ 更新时的时间戳

Link

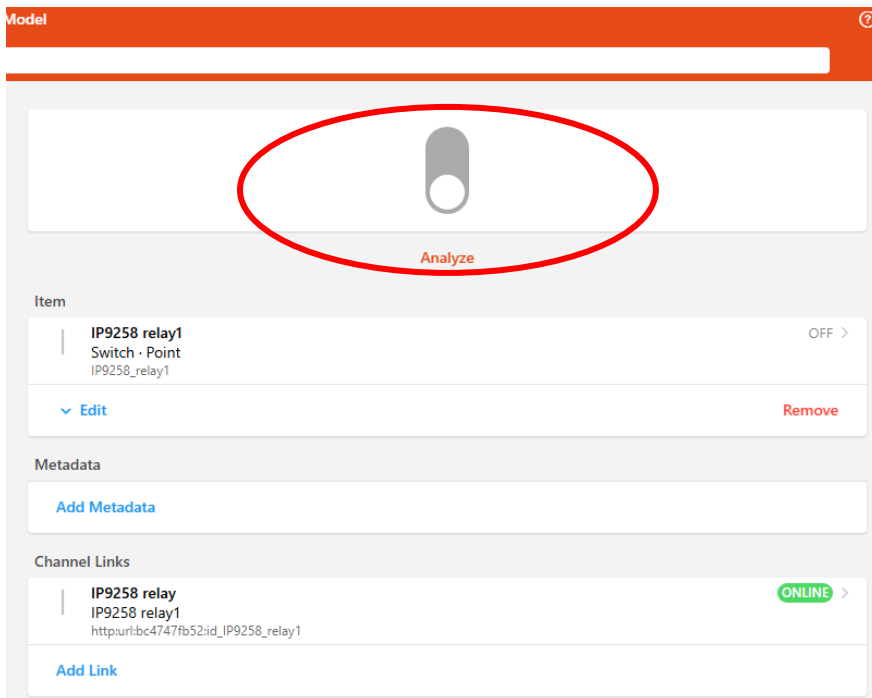
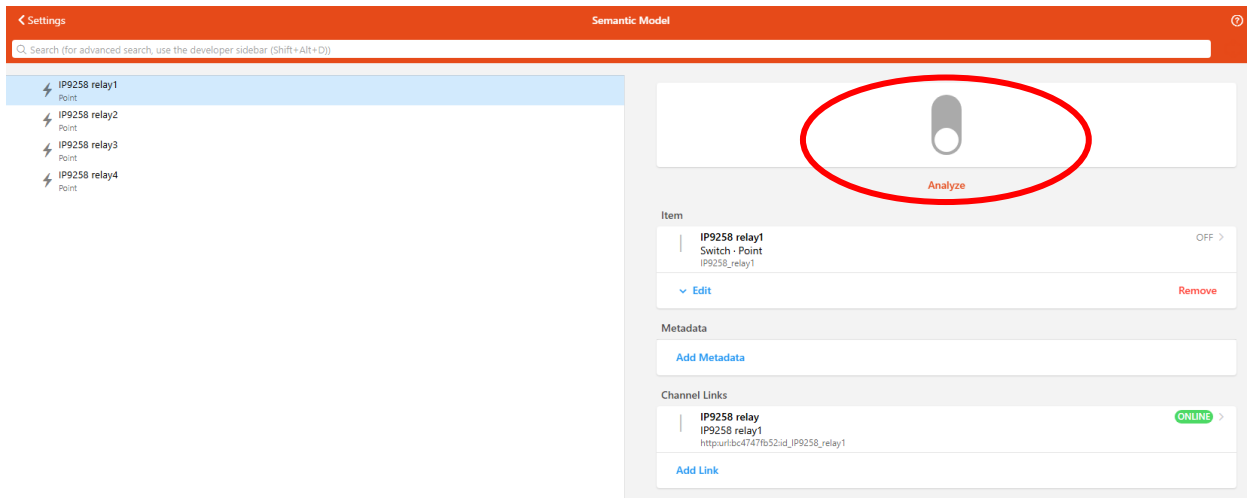
Once completed, it will display as shown in the image:



In next step, return to the main screen and click on the model on the left side.



After clicking, you will be taken to the Semantic Model. Once on that page, please click on any relay and then click on the Switch on the right. If your IP9258 is successfully triggered through openHAB, it means you have successfully set up the IP9258 configuration in openHAB.



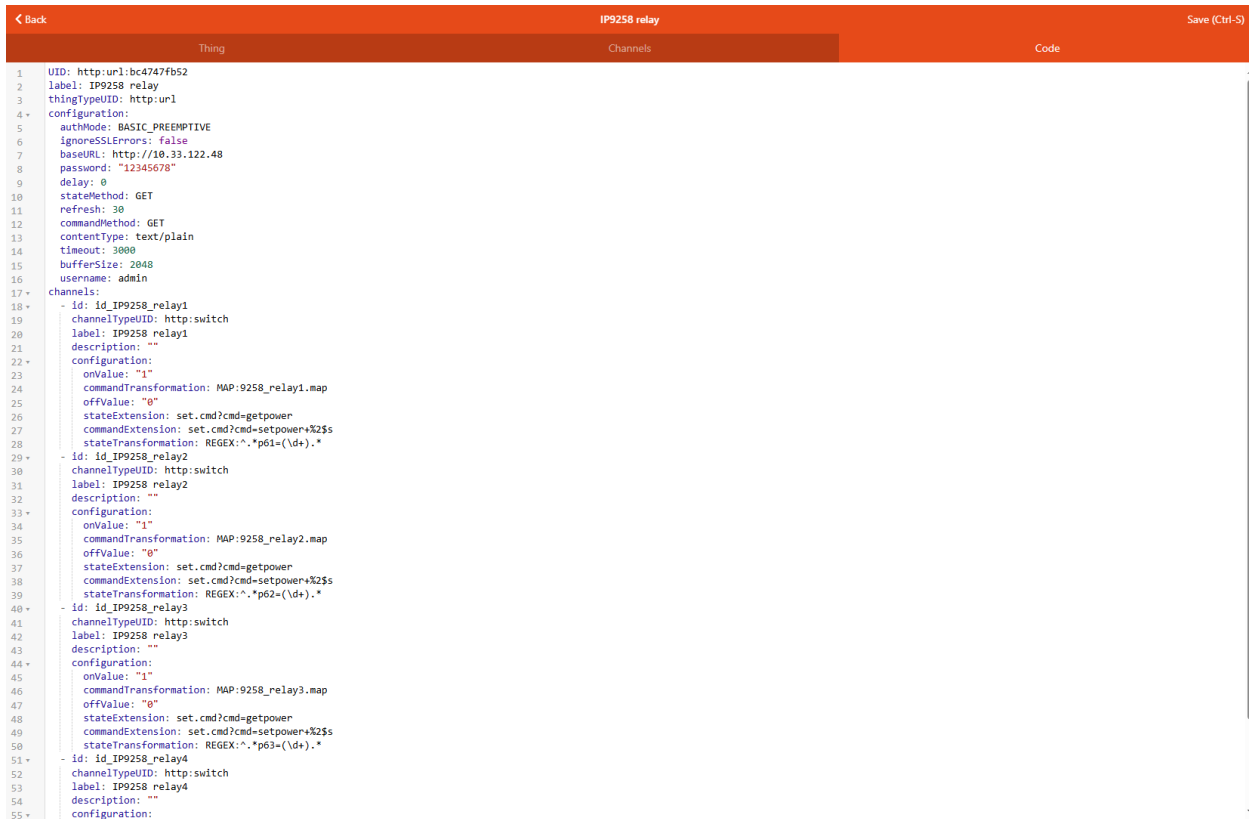
2.4 Configuration Parameters Corresponding to the 9258 Series Code

After adding all corresponding items to the four channels, you can view the parameters configured through the UI interface on the Code page.

You can also directly modify and add new channels on the Code page. If there are any errors, modifying the code directly, openHAB will provide a prompt.

Remember to click the Save button (Ctrl-S) in the top right corner to save the changes. The channels and Things will be updated immediately.

***Note:** It is recommended to check the code every time you configure Thing and Channel, as the user added or modified Things and Channels may not always be synchronized.



```
1  UID: http://url:bc4747fb52
2  label: IP9258 relay
3  thingTypeUID: http://url
4  configuration:
5    authMode: BASIC_PREAMPTIVE
6    ignoreSSLErrors: false
7    baseURL: http://10.33.122.48
8    password: "12345678"
9    delay: 0
10   stateMethod: GET
11   refresh: 30
12   commandMethod: GET
13   contentType: text/plain
14   timeout: 3000
15   bufferSize: 2048
16   username: admin
17  channels:
18  - id: id_IP9258_relay1
19    channelTypeUID: http://switch
20    label: IP9258_relay1
21    description: ""
22    configuration:
23      onValue: "1"
24      commandTransformation: MAP:9258_relay1.map
25      offValue: "0"
26      stateExtension: set.cmd?cmd=getpower
27      commandExtension: set.cmd?cmd=setpower+%2$s
28      stateTransformation: REGEX:~.*p61=(\d+).~*
29  - id: id_IP9258_relay2
30    channelTypeUID: http://switch
31    label: IP9258_relay2
32    description: ""
33    configuration:
34      onValue: "1"
35      commandTransformation: MAP:9258_relay2.map
36      offValue: "0"
37      stateExtension: set.cmd?cmd=getpower
38      commandExtension: set.cmd?cmd=setpower+%2$s
39      stateTransformation: REGEX:~.*p62=(\d+).~*
40  - id: id_IP9258_relay3
41    channelTypeUID: http://switch
42    label: IP9258_relay3
43    description: ""
44    configuration:
45      onValue: "1"
46      commandTransformation: MAP:9258_relay3.map
47      offValue: "0"
48      stateExtension: set.cmd?cmd=getpower
49      commandExtension: set.cmd?cmd=setpower+%2$s
50      stateTransformation: REGEX:~.*p63=(\d+).~*
51  - id: id_IP9258_relay4
52    channelTypeUID: http://switch
53    label: IP9258_relay4
54    description: ""
55    configuration:
```

To reference the code for the Things and Channels configuration in this image, please refer to P.51.

3. Setting IP Power for 9850, 9858MT, 9855 series

3.1 Create things

Similar to the setting method in 2.1, choose HTTP Binding during setup. Click on HTTP URL Thing to create a new thing named http_98XX_relay.

Fill in the Base URL field with the IP address of the 98XX device; Username and Password fields with the user and password of IP98XX; Refresh Time field with the refresh time for reading the on/off status of the 98XX device, defaulting to 30 seconds; choose GET for State Method, and POST for Command Method; select application/json for Content Type.

For filling the Configuration, refer to the image below (Ex model: IP POWER 9850).

The screenshot shows the configuration page for a device named 'http_9850_relay'. The page has a header with the device name and a 'Channels' button. Below the header, there is a table with fields: Identifier (http:url:9b4e47dc4d), Label (http_9850_relay), and Location (e.g. Kitchen). Under the 'Information' section, the 'Thing Type' is set to 'HTTP URL Thing'. The 'Configuration' section is expanded, showing a 'Show advanced' checkbox that is checked. The configuration fields are as follows:

| Field | Value | Status |
|--------------|---------------------|-------------|
| Base URL | http://10.33.122.52 | ✗ (Error) |
| Refresh Time | 30 | ✓ (Success) |
| Timeout | 3000 | ✓ (Success) |
| Delay | 0 | ✓ (Success) |
| Buffer Size | 2048 | ✓ (Success) |
| Username | admin | ⚠ (Warning) |
| Password | | ⚠ (Warning) |

Additional notes and descriptions:

- Base URL:** *Before you filling, Please refer to P.16 notice. Required The URL set here can be extended in the channel configuration.
- Refresh Time:** Time between two refreshes of all channels.
- Timeout:** The timeout in ms for each request.
- Delay:** Delay between to requests.
- Buffer Size:** Size of the response buffer (default 2048 kB).
- Username:** Basic Authentication username.
- Password:** Basic Authentication password.

***Note:** Except for the 9855 series which requires Preemptive Basic Authentication, other 98 series only need to click basic authentication to connect and use.

http_9850_relay

Channels

Authentication Mode

☒ Basic Authentication

☐ Preemptive Basic Authentication

☐ Digest Authentication

State Method

☒ GET

☐ POST

☐ PUT

HTTP method (GET,POST, PUT) for retrieving a status.

Command Method

☐ GET

☒ POST

☐ PUT

HTTP method (GET,POST, PUT) for sending commands.

Content Type

☒ application/json

☐ application/xml

☐ text/html

☐ text/plain

☐ text/xml

The MIME content type. Only used for 'POST' and 'PUT'.

Fallback Encoding

Fallback Encoding text received by this thing's channels.

Headers

Additional headers send along with the request

Ignore SSL Errors

☐

If set to true ignores invalid SSL certificate errors. This is potentially dangerous.

27

3.2 Create Channels

Similar to section 2.2, create a channel with an ID named `http_9850_relay1`. And prepare four map files (`http_cmd1.map` to `http_cmd4.map`) as mappings for the commands of four relays. These files should be saved in the installation directory under “`openHAB\conf\transform`”.

For example, the content of `http_cmd1.map` is as follows:

```
1= {"cmd":"setpower", "RL":[{"id":1, "action":1}]}
```

```
0= {"cmd":"setpower", "RL":[{"id":1, "action":0}]}
```

In JSON format, “`id`”:1 represents the first relay, and “`action`”:1 represents activation (turning on).

Add Channel
http_9850_relay

Channel

✓

Channel Identifier

id_http_9850_relay1

Note: cannot be changed after the creation

✓

Label

http 9850 relay1

✕

Description

Channel type

☐ Color Channel

☐ Contact Channel

☐ DateTime Channel

☐ Dimmer Channel

☐ Image Channel

☐ Location Channel

☐ Number Channel

☐ Player Channel

☐ Rollershutter Channel

☐ String Channel

☒ Switch Channel

✓

Please fill in the “state transformation” according to the instructions below:
JSONPATH: \$.result.RL[??].state

EX:

JSONPATH:\$.result.RL[0].state → Relay 1

JSONPATH:\$.result.RL[1].state → Relay 2

JSONPATH:\$.result.RL[2].state → Relay 3

·
·

Please adjust based on the number of your model ports.

Configuration

Show advanced ☒

State Transformation
JSONPATH:\$.result.RL[0].state

Transformation pattern used when receiving values. Chain multiple transformations with the mathematical intersection character "∩".

Command Transformation
MAP:http_cmd1.map

Transformation pattern used when sending values. Chain multiple transformations with the mathematical intersection character "∩".

State URL Extension
json.cmd?getpower

This value is added to the base URL configured in the thing for retrieving values.

Command URL Extension
json.cmd?

This value is added to the base URL configured in the thing for sending values.

After confirming that the settings match those in the picture, click "create," and the setup for Channel 9850 will be completed.

On Value
1

Required The value that represents ON

Off Value
0

Required The value that represents OFF

Read/Write Mode

☒ Read/Write

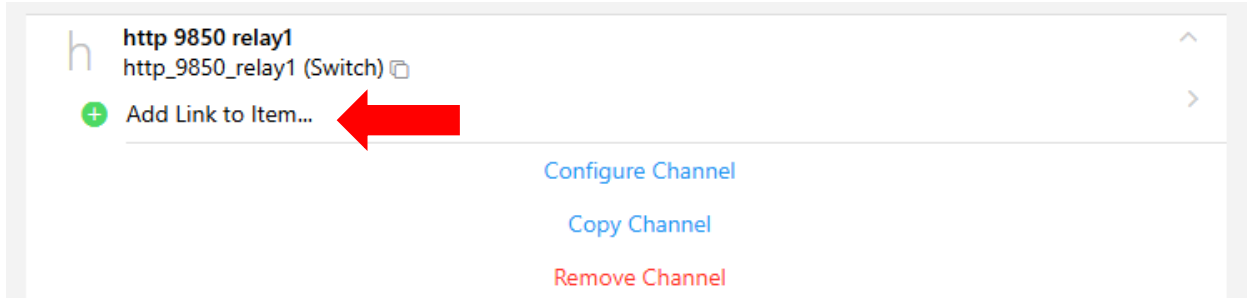
☐ Read Only

☐ Write Only

Create

3.3 Create item

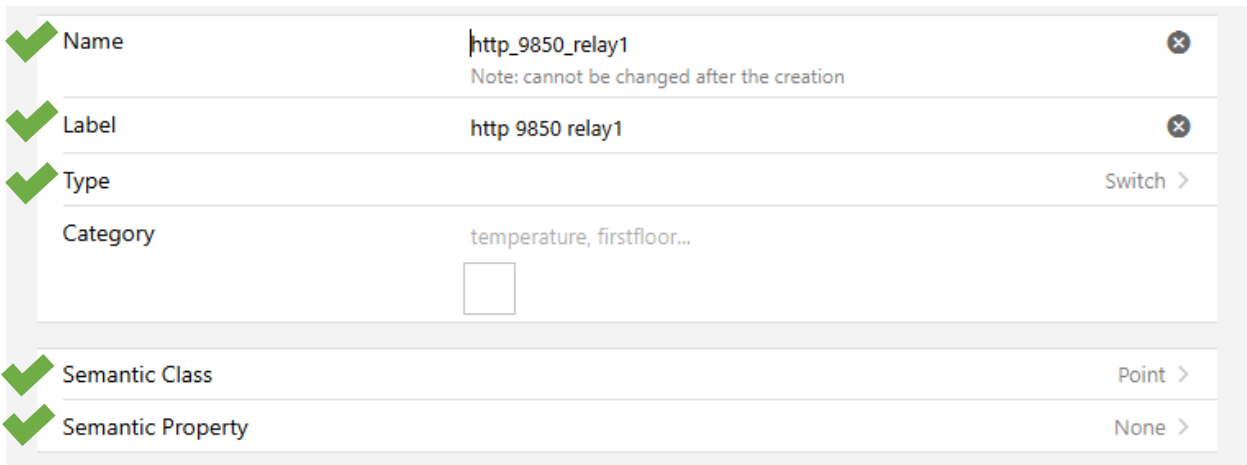
Back to the channel of IP9850, create a new item. The creation process is the same as in section 2.2. Ensure consistency with the settings shown in the image.



h http 9850 relay1
http_9850_relay1 (Switch)

+ Add Link to Item...

Configure Channel
Copy Channel
Remove Channel



✓ Name http_9850_relay1
Note: cannot be changed after the creation

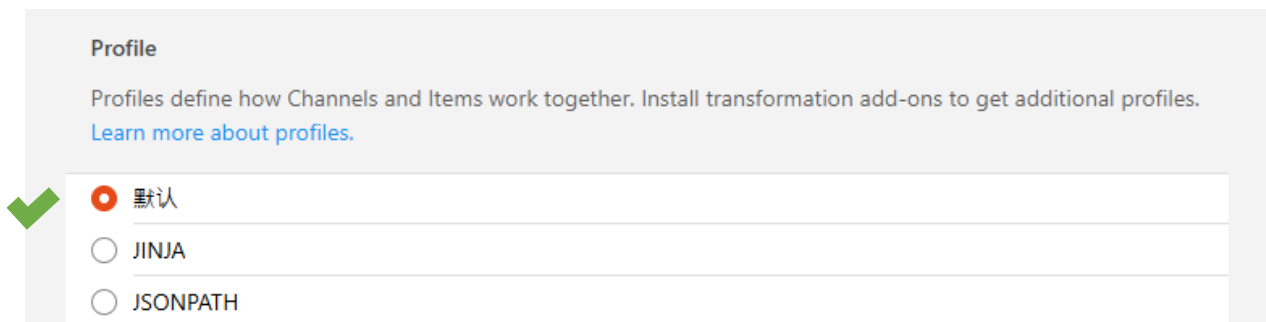
✓ Label http 9850 relay1

✓ Type Switch >

Category temperature, firstfloor...

✓ Semantic Class Point >

✓ Semantic Property None >



Profile

Profiles define how Channels and Items work together. Install transformation add-ons to get additional profiles.
[Learn more about profiles.](#)

✓ ☒ 默认

☐ JINJA

☐ JSONPATH

3.4 Configuration Parameters Corresponding to the 98 Series Code

The inquiry and modification methods are the same as in section 2.3. For reference to the configuration codes of the IP9850 series manual, please refer to P.52.

3.5 Create attributes Channel

Some models of IP POWER have functions to detect temperature, current, and voltage. To detect these attributes, we need to create an additional channel on the IP POWER models that have these functionalities.

Example model: IP POWER 9855 PRO

After creating a new Thing for the IP POWER, establish a channel within it.

Link Channel to Item

Channel
http 9855 PRO temperature
http:url:c826bee367:http_9855_PRO_temperature (Number)

Item
☐ Use an existing Item
☒ Create a new Item

| | | |
|--|----------------------------|---|
| Name | http_9855_PRO_temperature | ✕ |
| Note: cannot be changed after the creation | | |
| Label | http 9855 PRO temperature | ✕ |
| Type | Number | > |
| Dimension | | > |
| Category | temperature, firstfloor... | |

| | | |
|-------------------|-------|---|
| Semantic Class | Point | > |
| Semantic Property | None | > |

| | |
|-------------------|---|
| Non-Semantic Tags | ▼ |
|-------------------|---|

| | |
|-----------------|---|
| Parent Group(s) | > |
|-----------------|---|


Profile
Profiles define how Channels and Items work together. Install transformation add-ons to get additional profiles.
[Learn more about profiles.](#)

☒ 默认
☐ JINJA
☐ JSONPATH

Except for the Channel Identifier and Label, directly replicate the other settings as shown in the image.

Add Channel
http_9855_PRO_relay

Channel

| | |
|--------------------|---|
| Channel Identifier | http_9855_PRO_temperature <small>Note: cannot be changed after the creation</small> |
| Label | http 9855 PRO temperature  |
| Description | |

Channel type

☐ Color Channel

☐ Contact Channel

☐ DateTime Channel

☐ Dimmer Channel

☐ Image Channel

☐ Location Channel

☒ Number Channel

☐ Player Channel

☐ Rollershutter Channel

☐ String Channel

☐ Switch Channel

After completing the Configuration settings, click on "create" to finish.

*** Note:** In the State Transformation and unit sections in the image, the property being set is "temperature." If setting a different property, modify the red text in the code below:

Ex: JINJA:{{value_json.result.global_measure.temperature}}

e.g: temperature → voltage.

e.g. for Unit: °C (temperature), AC (voltage), A (current)

***Some settings may need to be adjusted due to different properties, please refer to P.55 for guidance. (Vital)**

Configuration

Show advanced ☒

State Transformation

JINJA:{{value_json.result.global_measure.temperature}}

Transformation pattern used when receiving values. Chain multiple transformations with the mathematical intersection character "∩".

Command Transformation

Transformation pattern used when sending values. Chain multiple transformations with the mathematical intersection character "∩".

State URL Extension

json.cmd?getpower

This value is added to the base URL configured in the thing for retrieving values.

Command URL Extension

This value is added to the base URL configured in the thing for sending values.

Escaped URL

☐

This specifies whether the URL is already escaped. Applies to the base URL, commandExtension and stateExtension.

State Content

Content for state request (only used if method is POST/PUT)

Read/Write Mode

☐ Read/Write

☒ Read Only

☐ Write Only

Unit

°C

Unit to append to the (transformed) value.

After completing the Channel, proceed to create an item. Generally, no modifications are needed; it should match the settings in the image.

Once you confirm the settings are correct, create the item.

Link Channel to Item

Channel

http 9855 PRO temperature
http:url:c826bee367:http_9855_PRO_temperature (Number)

Item

☐ Use an existing Item

☒ Create a new Item

| | | |
|-----------|--|---|
| Name | http_9855_PRO_temperature | ✕ |
| | Note: cannot be changed after the creation | |
| Label | http 9855 PRO temperature | ✕ |
| Type | Number > | |
| Dimension | > | |
| Category | temperature, firstfloor... <div></div> | |

| | |
|-------------------|---------|
| Semantic Class | Point > |
| Semantic Property | None > |

| | |
|-------------------|---|
| Non-Semantic Tags | ▼ |
|-------------------|---|

☰ Parent Group(s)

>

Profile

Profiles define how Channels and Items work together. Install transformation add-ons to get additional profiles.
[Learn more about profiles.](#)

☒ 默认

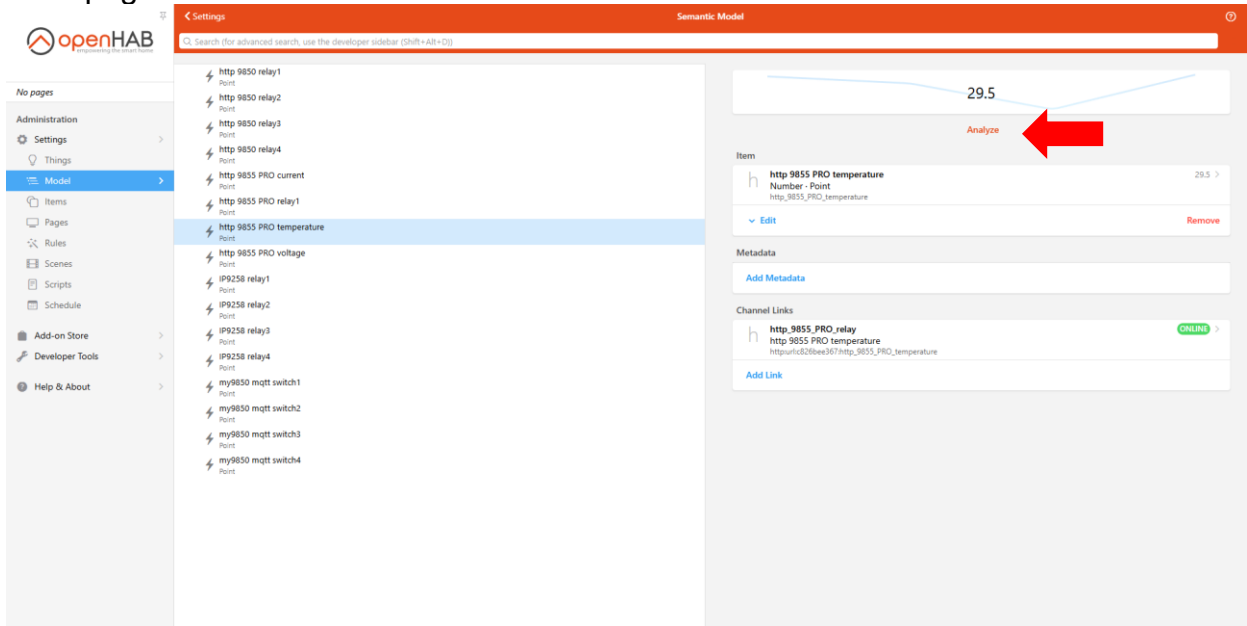
☐ JINJA

☐ JSONPATH

☐ MAP

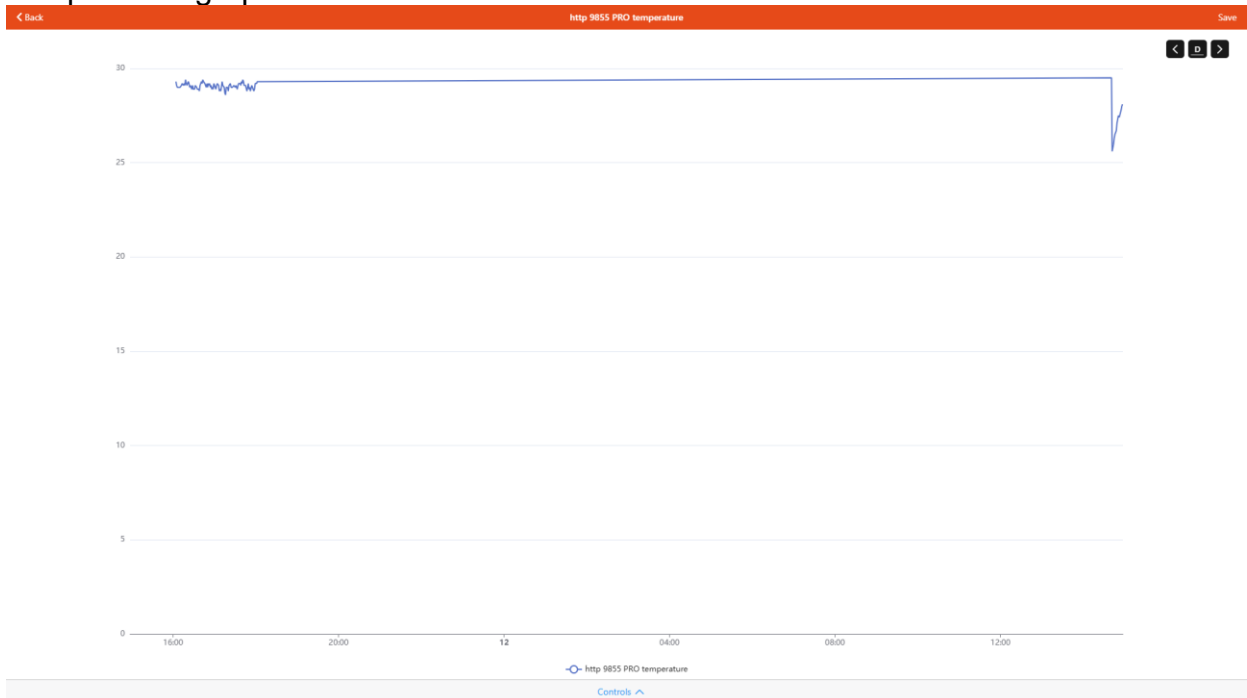
After completing the setup, it can be displayed in the model, and clicking on "Analyze" on the right will show the graph.

Main page:



The screenshot shows the openHAB Semantic Model interface. The left sidebar contains a navigation menu with options like Administration, Settings, Things, Model, Items, Pages, Rules, Scenes, Scripts, Schedule, Add-on Store, Developer Tools, and Help & About. The 'Model' option is selected. The main area displays a list of items, including 'http 9850 relay1', 'http 9850 relay2', 'http 9850 relay3', 'http 9850 relay4', 'http 9855 PRO current', 'http 9855 PRO relay1', 'http 9855 PRO temperature' (highlighted), 'http 9855 PRO voltage', 'IP9258 relay1', 'IP9258 relay2', 'IP9258 relay3', 'IP9258 relay4', 'my9850 mqtt switch1', 'my9850 mqtt switch2', 'my9850 mqtt switch3', and 'my9850 mqtt switch4'. On the right, the details for the selected item 'http 9855 PRO temperature' are shown, including a value of 29.5 and an 'Analyze' button. A red arrow points to the 'Analyze' button.

Temperature graph:



4. MQTT setting

This allows users to connect and control IP Power devices outside without connecting the intranet. (Currently only supports IP Power 98 series)

(4-1) First, please turn on your IP POWER device and enter Application→IP Service.

Select CNT(MQTT) Setting, and click MQTT Service, select Custom.

The image shows two screenshots of the 'CNT(MQTT) Setting (Remote control)' interface. The top screenshot shows the 'MQTT Service' dropdown menu set to 'IPPower', with a red arrow pointing to it. The bottom screenshot shows the 'MQTT Service' dropdown menu set to 'Custom', with a red arrow pointing down from the top screenshot. The bottom screenshot also shows additional fields for Server, Port, User, Password, SSL/TLS, and Enable Server Certificate.

| CNT(MQTT) Setting (Remote control) | |
|------------------------------------|--------------------------|
| MQTT Service | IPPower |
| Enable | <input type="checkbox"/> |
| Status | |
| <button>Apply</button> | |

| CNT(MQTT) Setting (Remote control) | |
|------------------------------------|--------------------------|
| MQTT Service | Custom |
| Enable | <input type="checkbox"/> |
| Server : | |
| Port : | |
| User : | |
| Password : | |
| SSL/TLS | <input type="checkbox"/> |
| Enable Server Certificate | <input type="checkbox"/> |
| Status | |
| <button>Apply</button> | |

(4-2) If you already have a dedicated MQTT broker, please fill in the information, SSL/TLS and Certificate should be configured according to user requirements, but remember to click "Enable" After setting, please click "Apply". If "Connect" appears in Status after clicking, it means it is successfully activated.

If you don't have your own MQTT broker, you can choose from some online free MQTT brokers.

For example: broker.emqx.io. Please do not choose test.mosquitto.org because it has compatibility issues with our devices.

When selecting a public broker, please try to use TLS secure connection on port 8883 as it will be exposed to all platforms integrated with this MQTT broker, so please be mindful of security risks.

*The tutorial will use port 1883 as the configuration example.

*If you want to use our company's IP POWER as MQTT Broker, please contact with us.

4.1 Create MQTT Broker

Click MQTT Binding → then click MQTT Broker.

The screenshot shows two parts of the configuration interface. The top part is a list of bindings: 'binding-exec' with 'Exec Binding', 'binding-http' with 'HTTP Binding', and 'binding-mqtt' with 'MQTT Binding'. Each item has a right-pointing arrow. The bottom part is titled 'Add Manually' and contains two items: 'mqttbroker' with 'MQTT Broker' and a description 'A connection to a MQTT broker', featuring a blue 'Bridge' button and a right arrow; and 'mqtttopic' with 'Generic MQTT Thing' and a description 'You need a configured Broker first. Dynamically add channels of various types to this Thing. Link different MQTT topics to each channel.', featuring a right arrow.

| | | |
|--------------|--------------|---|
| binding-exec | Exec Binding | > |
| binding-http | HTTP Binding | > |
| binding-mqtt | MQTT Binding | > |

Add Manually

| | | |
|--|--------------------|----------|
| mqttbroker | MQTT Broker | Bridge > |
| A connection to a MQTT broker | | |
| mqtttopic | Generic MQTT Thing | > |
| You need a configured Broker first. Dynamically add channels of various types to this Thing. Link different MQTT topics to each channel. | | |

Please configure according to the image below. If you are setting it up with your own broker, please change the Broker Hostname/IP.

The screenshot shows the 'MQTT Broker' configuration form. It has a table with fields for Identifier, Label, and Location. Below the table is the title 'MQTT Broker' and a description 'A connection to a MQTT broker'. There is a 'Show advanced' checkbox which is checked. The form contains several input fields: 'Broker Hostname/IP' with the value 'broker.emqx.io', 'Broker Port' with the value '1883', and 'Secure Connection' which is a toggle switch currently turned off. There are also labels for 'Required' fields. At the bottom, there is a 'Hostname Validated' toggle switch which is turned on.

| | |
|------------|-----------------------|
| Identifier | mqttbroker:8496c99244 |
| Label | MQTT Broker |
| Location | e.g. Kitchen |

MQTT Broker

A connection to a MQTT broker

Show advanced ☒

Broker Hostname/IP
broker.emqx.io

Required The IP/Hostname of the MQTT broker

Broker Port
1883

The port is optional, if none is provided, the typical ports 1883 and 8883 (SSL) are used.


Secure Connection ☐

Required Uses TLS/SSL to establish a secure connection to the broker.


Hostname Validated ☒

Validate hostname from certificate against server hostname for secure connection.

4.2 Create Generic MQTT Thing

Please follow the same setup as in section 4.12. On the 'Things' page, click on  then select MQTT Binding -> Generic MQTT Thing.


Once on the page, refer to the image below for editing. Please click on 'Create Thing' below.

| | |
|--|----------------------|
| Identifier  | mqtt:topice202ba2650 |
| Label | Generic MQTT Thing |
| Location | e.g. Kitchen |

Parent Bridge



This type of Thing needs to be associated to a working Bridge to function properly.

Note 1


Bridge 

Generic MQTT Thing

You need a configured Broker first. Dynamically add channels of various types to this Thing. Link different MQTT topics to each channel.


This is your device's MAC address  Show advanced 

Availability Topic

mac/[REDACTED]/status 


Topic of the LWT of the device

Device Available Payload

online 


Payload of the 'Availability Topic', when the device is available. Default: 'ON'

Device Unavailable Payload

offline 

Payload of the 'Availability Topic', when the device is *not* available. Default: 'OFF'

Availability Payload Transformations


JS:mqtt_availability.js 


Applies transformations to the incoming availability payload. A transformation example for a received JSON would be "JSONPATH:\$.status" for a json {status: "Online"}. You can chain transformations by separating them with the intersection character ^.

Note 1: Click on Bridge and select the MQTT Broker currently connected.

Parent Bridge

Bridge

☐ 


☒ MQTT Broker (mqtt:broker:8496c99244) 

4.3 Create Generic MQTT Thing Channels

Please refer to the settings in the image below to create Channels.

If you are setting up for the first time, it's recommended not to change the Identifier and Label names in the Channel except for IP Power. Keeping them aligned with P.54 reference codes will be more convenient.

Channel

| | |
|--------------------|---|
| Channel Identifier | my9850_mqtt_switch1 |
| | Note: cannot be changed after the creation |
| Label | my9850 mqtt switch1  |
| Description | |

Channel type

☐ Text Value

☐ Number Value

☐ Dimmer

☒ On/Off Switch

☐ Open/Close Contact

☐ Color Value (Red,Green,Blue)

☐ Color Value (Hue,Saturation,Brightness)

☐ Color Value (HSB, RGB or CIE xyY)

☐ Date/Time Value
Current date and/or time

☐ Image
An image to display. Send a binary bmp, jpg, png or any other supported format to this channel.

☐ Location
GPS coordinates as Latitude,Longitude,Altitude

☐ Rollershutter

☐ Trigger

Advanced setting 1

Notice 1: The part covered in black is where you should enter the device's MAC address.

Configuration

Show advanced ☒

Notice 1

MQTT State Topic

mac/[REDACTED]/cmd_ack/id_my_openhab

An MQTT topic that this thing will subscribe to, to receive the state. This can be left empty, the channel will be state-less command-only channel.

Notice 1

MQTT Command Topic

mac/[REDACTED]/cmd

An MQTT topic that this thing will send a command to. If not set, this will be a read-only switch.

QoS

☒ At most once (best effort delivery "fire and forget")

☐ At least once (guaranteed that a message will be delivered at least once)

☐ Exactly once (guarantees that each message is received only once by the counterpart)

MQTT QoS of this channel (0, 1, 2). Default is QoS of the broker connection.

Retained

☐

The value will be published to the command topic as retained message. A retained value stays on the broker and can even be seen by MQTT clients that are subscribing at a later point in time.

Is Command

☐

If the received MQTT value should not only update the state of linked items, but command them, enable this option.

Custom On/Open Value

p1_1

A number (like 1, 10) or a string (like "enabled") that is additionally recognised as on/open state. You can use this parameter for a second keyword, next to ON (OPEN respectively on a Contact).

Custom Off/Closed Value

p1_0

A number (like 0, -10) or a string (like "disabled") that is additionally recognised as off/closed state. You can use this parameter for a second keyword, next to OFF (CLOSED respectively on a Contact).

Advanced setting 2

Please pay attention to the section titled "Incoming Value Transformations". Due to the length of the script that needs to be entered in this section, please copy the paragraph below.

Incoming Value Transformations Script:

```
JINJA:"{% if (value_json.cmd == 'getpower' and value_json.result.RL[1].state == 1) or  
(value_json.tag == 'r1_on' and value_json.result.message == 'success') %} 'p1_1' {% elif  
(value_json.cmd == 'getpower' and value_json.result.RL[1].state == 0) or (value_json.tag ==  
'r1_off' and value_json.result.message == 'success')%} 'p1_0' {% else %} 'p1_0' {% endif %}"
```

If you need to create the same channels later, please modify the parts highlighted in red.

Ex:

1. r1_on → r2_on
2. P1_1 → P2_2
3. r1_off → r2_off
4. P1_0 → P2_0

Transform Values

These configuration parameters allow you to alter a value before it is published to MQTT or before a received value is assigned to an item.

Incoming Value Transformations

JINJA:"{% if (value_json.cmd == 'getpower' and value_json.result.RL[0].state == 1) or (value_json.tag == 'r1_on' and

Applies transformations to an incoming MQTT topic value. A transformation example for a received JSON would be "JSONPATH:\$device.status.temperature" for a json {device: {status: { temperature: 23.2 }}}. You can chain transformations by separating them with the intersection character ^.

Outgoing Value Transformation

MAP:mqtt_cmd.map

Applies a transformation before publishing a MQTT topic value. Transformations are specialised in extracting a value, but some transformations like the MAP one could be useful.

Outgoing Value Format

%s

Format a value before it is published to the MQTT broker. The default is to just pass the channel/item state. If you want to apply a prefix, say "MYCOLOR,", you would use "MYCOLOR,%s". If you want to adjust the precision of a number to for example 4 digits, you would use "%.4f".

After confirming that all settings in the image have been entered, press "Create" to complete the setup.

4.4 Create items

If you have completed the previous steps, create an item for the relay channel instructions from earlier. Click "Add Link to Item" under these channels, and create an Item based on the number of Channels you have created.

Except for the IP Power model shown in the picture, you don't need to change any other settings. Simply click "Link" to complete.

Channel

my9850 mqtt switch1
mqtt:topic202ba2650:my9850_mqtt_switch1 (Switch)

Item

☐ Use an existing Item

☒ Create a new Item

Name

Generic_MQTT_Thing_my9850_mqtt_switch1

Note: cannot be changed after the creation

Label

my9850 mqtt switch1

Type

Switch >

Category

temperature, firstfloor...

Semantic Class

Point >

Semantic Property

None >

Non-Semantic Tags

>

Parent Group(s)

>

Profile

Profiles define how Channels and Items work together. Install transformation add-ons to get additional profiles.
[Learn more about profiles.](#)

☒ 默认

☐ JINJA

☐ JSONPATH

☐ MAP

☐ REGEX

4.5 MQTT Code

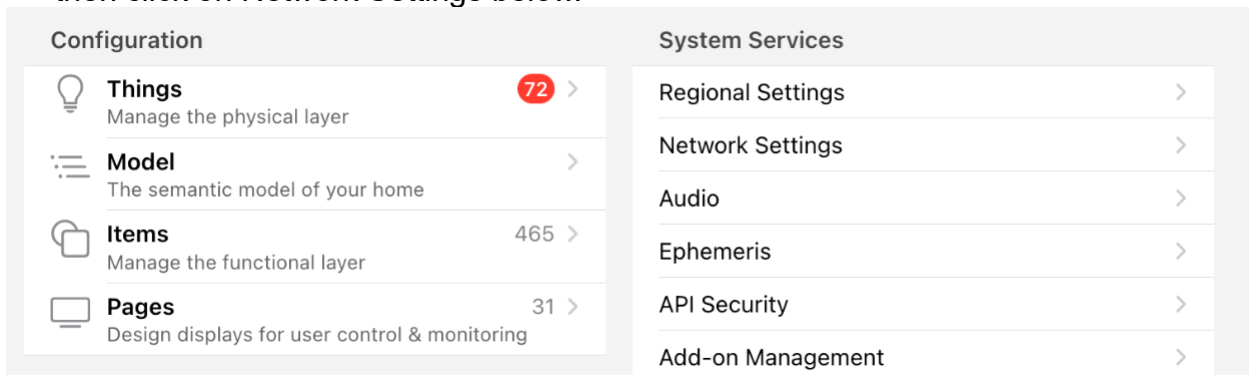
The inquiry and modification methods are the same as in section 2.3. For reference to the configuration codes of the MQTT, please refer to P.55

5. openHAB App LAN and WAN settings

openHAB has its own dedicated app for device control, but this app normally only supports local network control.

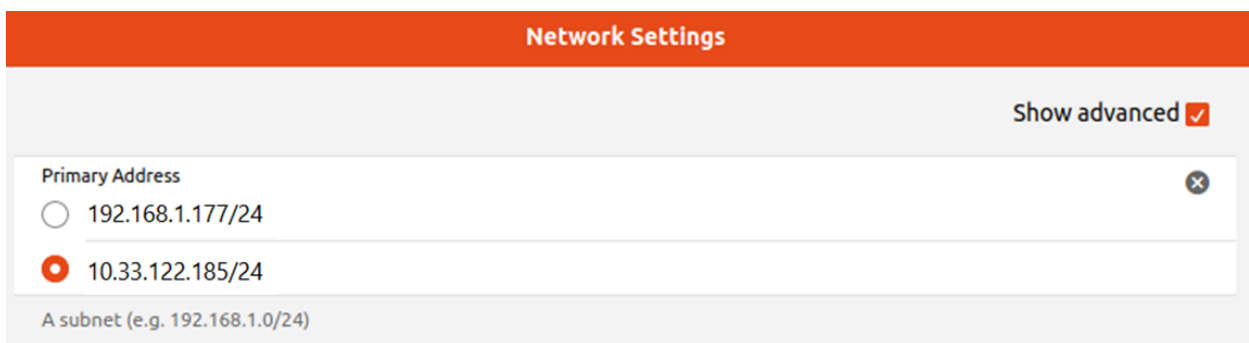
If you want to control your IP Power devices at home or work from outside using a smartphone or tablet, please follow these steps:

- (1) First, go to openHAB's Settings → System Services, then click on Network Settings below.



After clicking on it, you will see the Primary Address. Please select the IP you want to set up. Once selected, click the Save button at the top right.

*Note: This step is necessary if you want to control the device using the APP, as the newly installed, openHAB does not have a default IP.

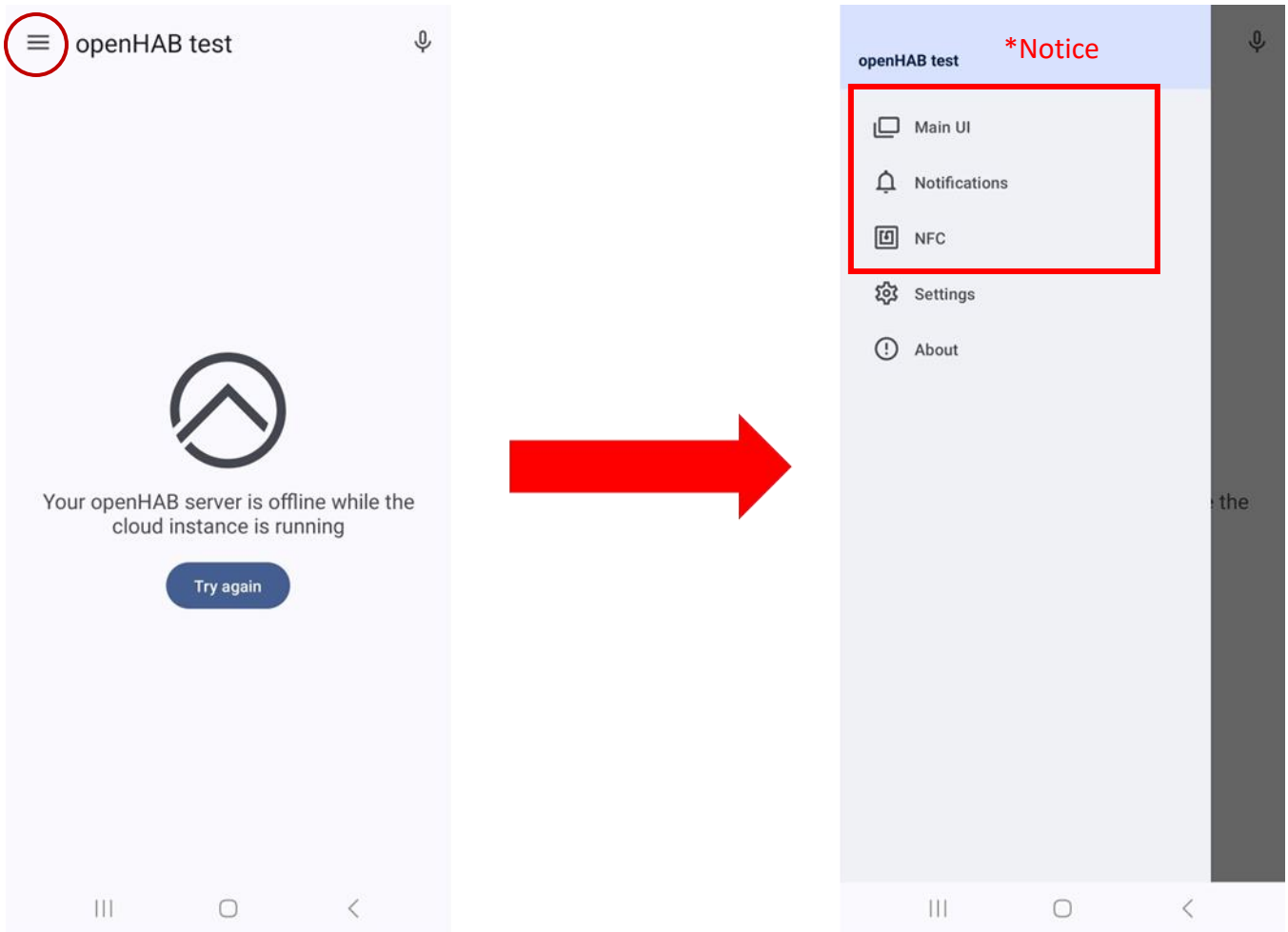


- (2) After completing the setup, please download the openHAB app on your mobile phone or tablet.

(3) After the download is complete, please open the openHAB mobile app and tap the icon inside the red circle in the image on the bottom left. When the screen showing the image on the right appears, tap "Settings."

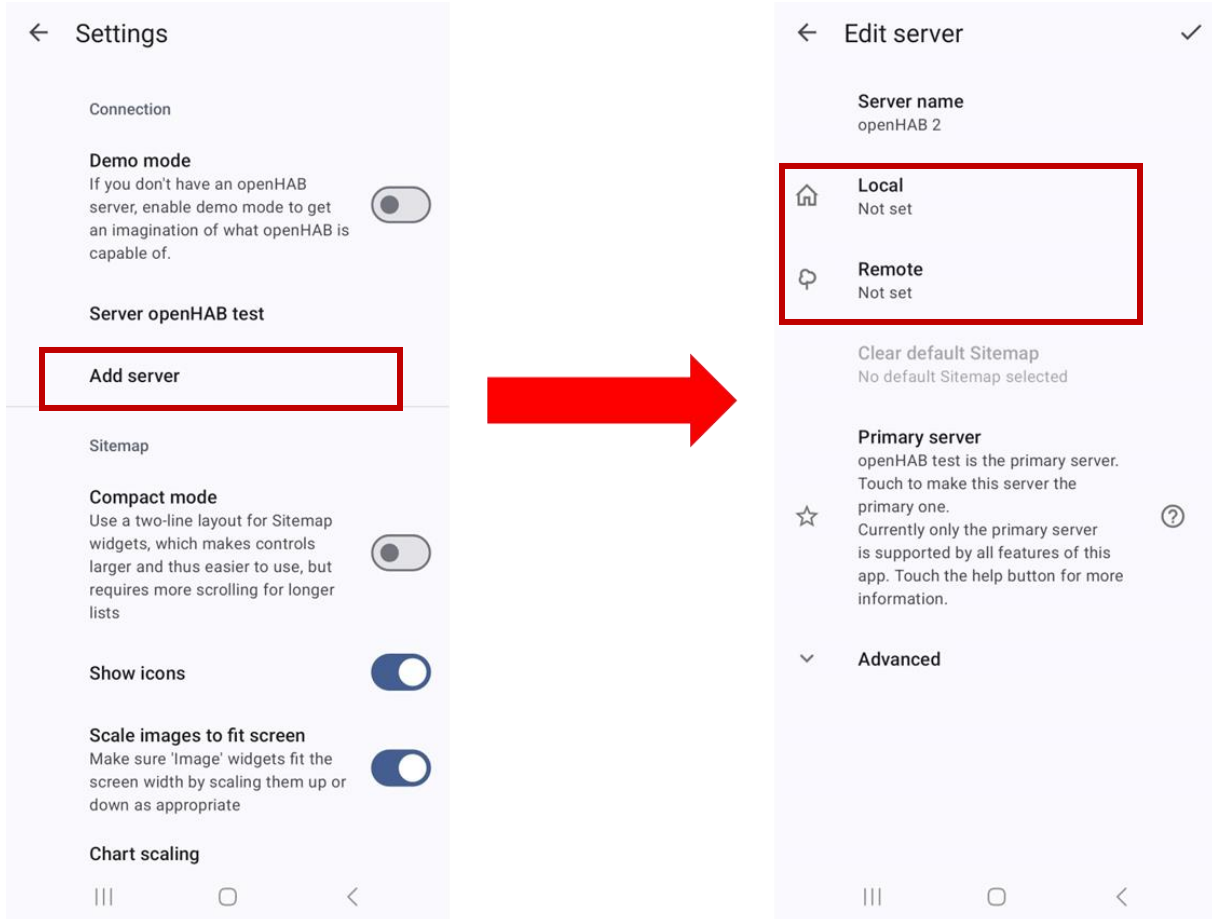
***Notice:** The option inside the red frame will not appear the first time you open openHAB.

Image example:



(4) After entering Settings, click Add server:

*To set up an internal network(Lan), click Local (recommended).
To set up an external network(Wan), click Remote.



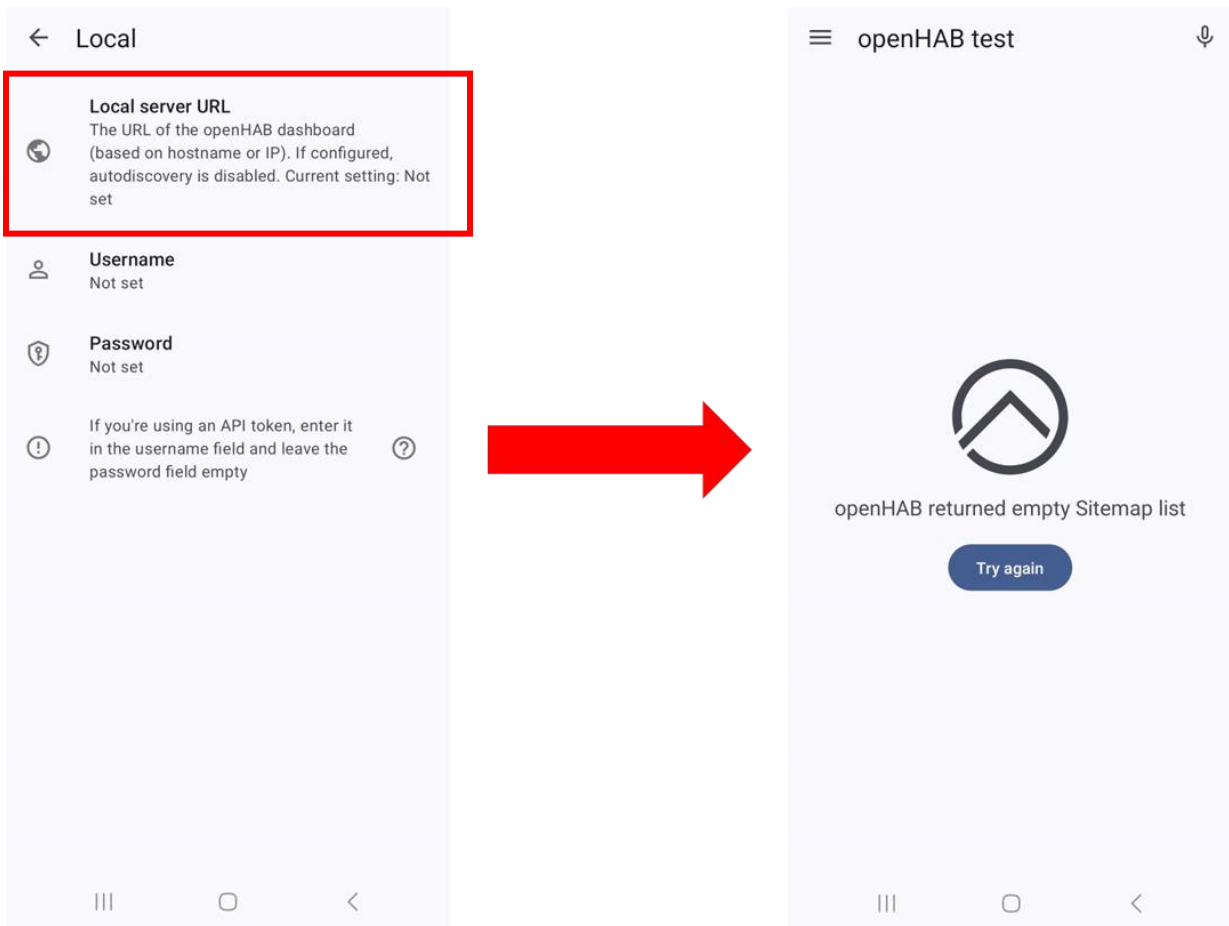
(5) Setting up Local Network (LAN) Connection:

Please click on the Local server URL inside the red box in the image, and enter the main location (IP) that you configured in openHAB.

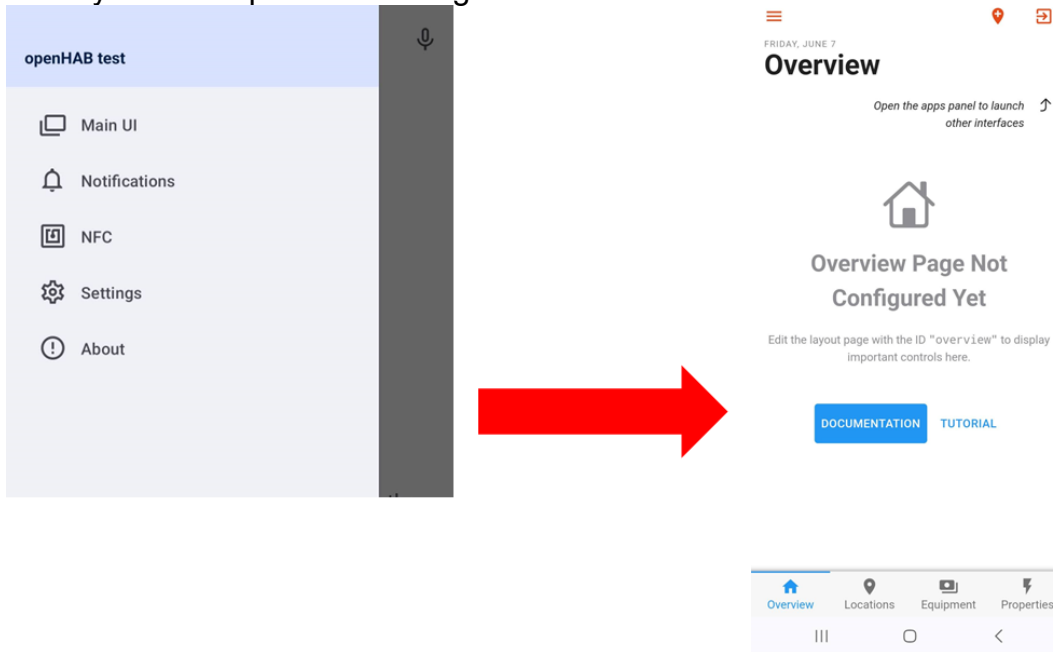
Please enter it in this format: `http://10.33.122.185:8080/`


***Notice:** Do not directly copy the red text; instead, paste the openHAB IP address you configured. (Refer to page P.43)

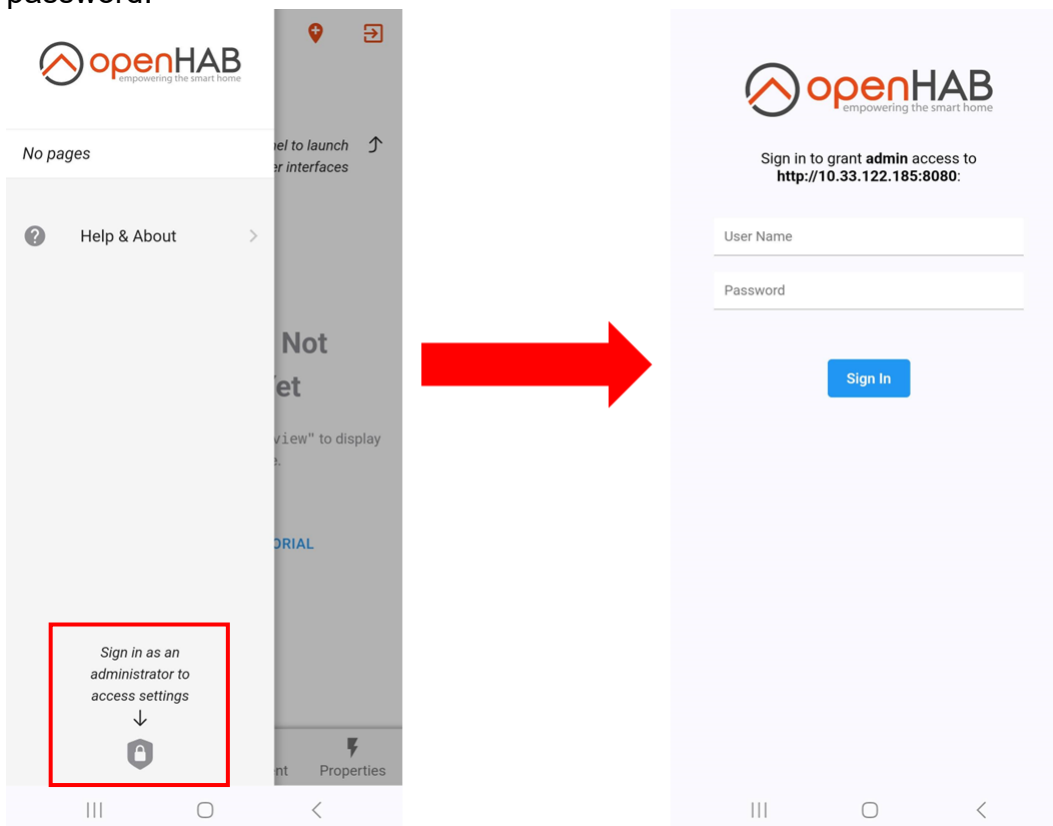
After configuration, return to the main screen to check if the connection is successful. If configured correctly, the main screen should no longer display "Your openHAB server is offline while the cloud instance is running."



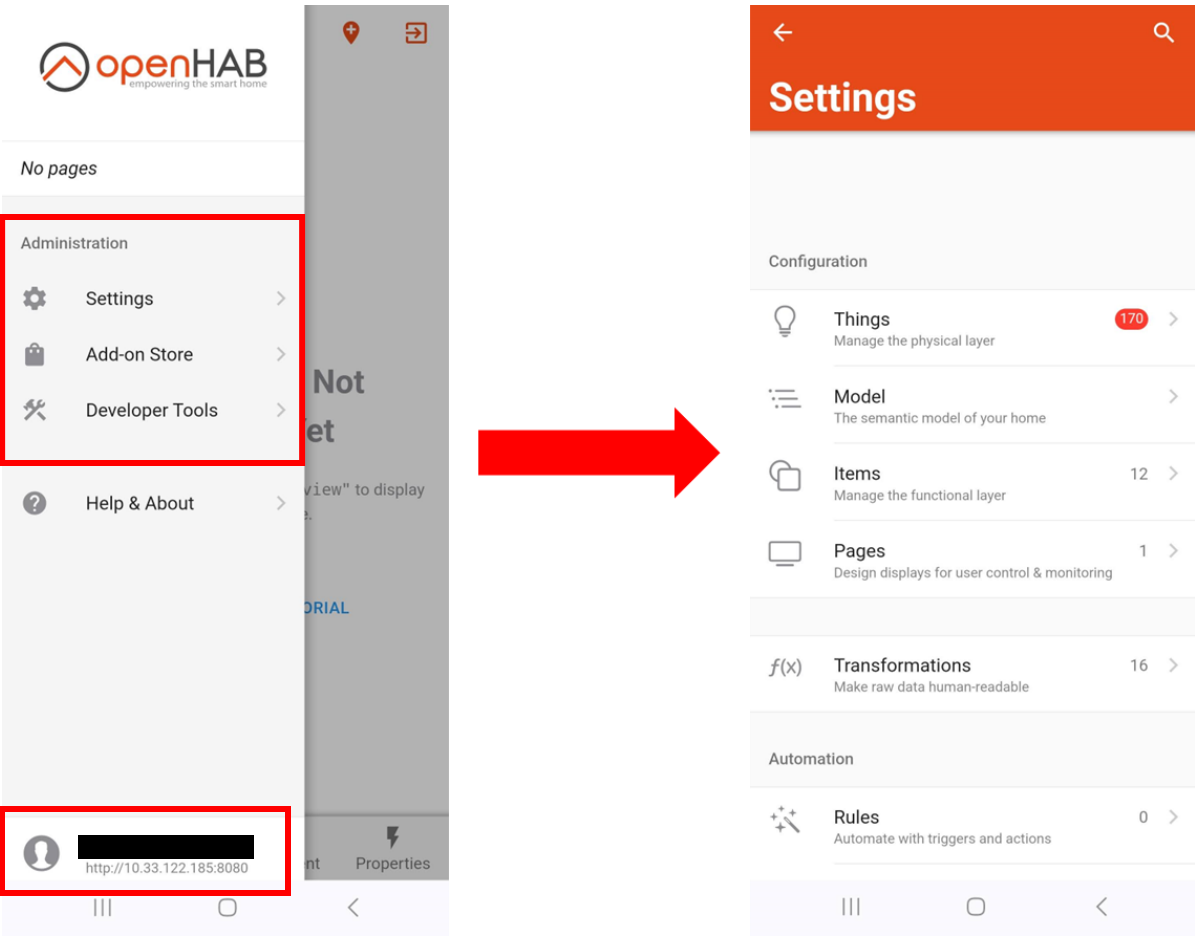
After returning to the main screen, swipe left and tap on "Main UI" to access the openHAB system interface (shown on the right). However, this does not mean you can directly use the openHAB settings.



To proceed, tap  in the top left corner of the screen, and then click on "Sign in" in the red box at the bottom left. Once on the login screen, enter your username and password.



After successfully logging in, you can use your personal settings. You can also log out at any time to switch to another user.

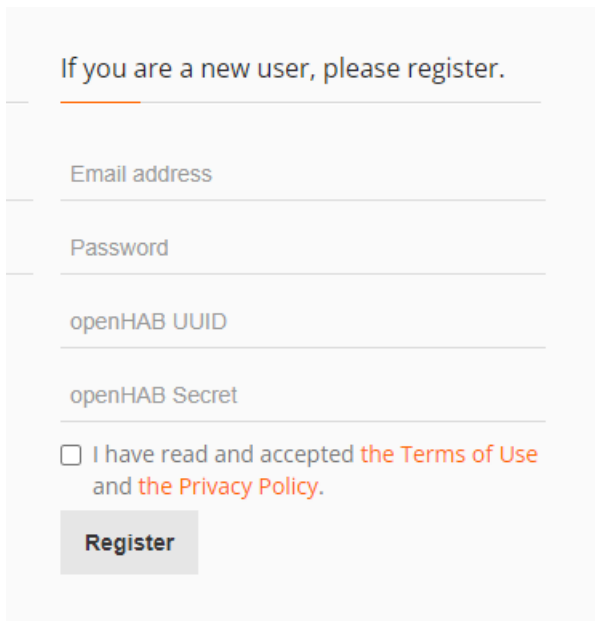


(6) Setting up Remote Teaching:

- *Notice:** 1. Please first install the openHAB Cloud Connector from the Add-on Store in openHAB. Remote control will not be possible without this feature.
2. It is recommended to complete the local network setup before proceeding with this configuration.

(6-1) Please start by accessing the website and registering an account.

During registration, you will be asked to provide the **UUID** and **Secret** of openHAB.



The image shows a registration form with the following fields and elements:

- A heading: "If you are a new user, please register."
- An "Email address" input field.
- A "Password" input field.
- An "openHAB UUID" input field.
- An "openHAB Secret" input field.
- A checkbox labeled "I have read and accepted the Terms of Use and the Privacy Policy."
- A "Register" button.

How to find UUID and Secret?

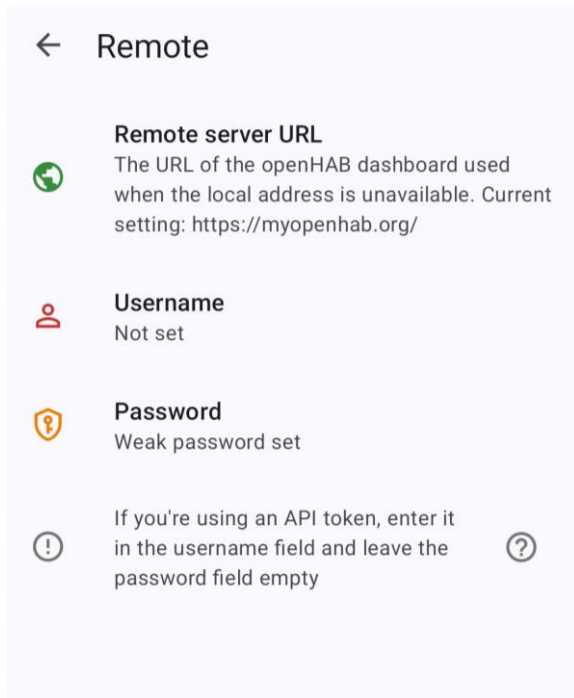
Please open your openHAB folder and follow these instructions:

1. **UUID:** Once inside your openHAB folder, navigate to 'userdata'. Inside, there is a file named UUID. Open this file with Notepad to find your UUID.
2. **Secret:** Similarly, within 'userdata', you will find a file named 'openhabcloud'. Inside this file, there is another file named 'secret'. Open this 'secret' file with Notepad to find your secret.

After registration, you will receive a verification email in your inbox. Once you confirm receive the mail and click on the verification link, your account registration will be complete.

Now let's return to the openHAB app. Please tap on "Settings" and then go to "Remote."

In the "Remote server URL" field, enter "<https://myopenhab.org>" (official appoint domain).



The username and password are the ones you used to register with openHAB.

Once done, please test the connection using mobile roaming mode or another wireless network. If you can connect successfully, it means you can remotely control devices inside openHAB from outside.

6. IP Power's code in openHAB

6.1 9258 code

UID: http:url:bc4747fb52

label: IP9258 relay

thingTypeUID: http:url

configuration:

authMode: BASIC_PREEMPTIVE

ignoreSSLErrors: false

baseURL: http://10.33.122.48

password: "12345678"

delay: 0

stateMethod: GET

refresh: 30

commandMethod: GET

contentType: text/plain

timeout: 3000

bufferSize: 2048

username: admin

channels:

- id: id_IP9258_relay1

channelTypeUID: http:switch

label: IP9258 relay1

description: ""

configuration:

onValue: "1"

commandTransformation: MAP:9258_relay1.map

offValue: "0"

stateExtension: set.cmd?cmd=getpower

commandExtension: set.cmd?cmd=setpower+%2\$s

stateTransformation: REGEX:^\.*p61=(\d+).*

- id: id_IP9258_relay2

channelTypeUID: http:switch

label: IP9258 relay2

description: ""

configuration:

onValue: "1"

commandTransformation: MAP:9258_relay2.map

offValue: "0"

stateExtension: set.cmd?cmd=getpower

commandExtension: set.cmd?cmd=setpower+%2\$s

stateTransformation: REGEX:^\.*p62=(\d+).*

- id: id_IP9258_relay3

channelTypeUID: http:switch

label: IP9258 relay3

```

description: ""
configuration:
  onValue: "1"
  commandTransformation: MAP:9258_relay3.map
  offValue: "0"
  stateExtension: set.cmd?cmd=getpower
  commandExtension: set.cmd?cmd=setpower+%2$s
  stateTransformation: REGEX:.*p63=(\d+).*
```

- id: id_IP9258_relay4

```

channelTypeUID: http:switch
label: IP9258 relay4
description: ""
configuration:
  onValue: "1"
  commandTransformation: MAP:9258_relay4.map
  offValue: "0"
  stateExtension: set.cmd?cmd=getpower
  commandExtension: set.cmd?cmd=setpower+%2$s
  stateTransformation: REGEX:.*p64=(\d+).*
```

6.2 98 series code

UID: http:url:9b4e47dc4d

label: http_98XX_relay

thingTypeUID: http:url

configuration:

```

authMode: BASIC
ignoreSSLErrors: false
baseURL: http://10.33.122.52
password: "12345678"
delay: 0
stateMethod: GET
refresh: 30
commandMethod: POST
contentType: application/json
timeout: 3000
bufferSize: 2048
username: admin
```

channels:

```

- id: http_9850_relay1
  channelTypeUID: http:switch
  label: http 9850 relay1
  description: ""
  configuration:
    onValue: "1"
    commandTransformation: MAP:http_cmd1.map
```

```

    offValue: "0"
    stateExtension: json.cmd?getpower
    commandExtension: json.cmd?
    stateTransformation: JSONPATH:$.result.RL[0].state
- id: http_9850_relay2
  channelTypeUID: http:switch
  label: http 9850 relay2
  description: ""
  configuration:
    onValue: "1"
    commandTransformation: MAP:http_cmd1.map
    offValue: "0"
    stateExtension: json.cmd?getpower
    commandExtension: json.cmd?
    stateTransformation: JSONPATH:$.result.RL[1].state
- id: http_9850_relay3
  channelTypeUID: http:switch
  label: http 9850 relay3
  description: ""
  configuration:
    onValue: "1"
    commandTransformation: MAP:http_cmd1.map
    offValue: "0"
    stateExtension: json.cmd?getpower
    commandExtension: json.cmd?
    stateTransformation: JSONPATH:$.result.RL[2].state
- id: http_9850_relay4
  channelTypeUID: http:switch
  label: http 9850 relay4
  description: ""
  configuration:
    onValue: "1"
    commandTransformation: MAP:http_cmd1.map
    offValue: "0"
    stateExtension: json.cmd?getpower
    commandExtension: json.cmd?
    stateTransformation: JSONPATH:$.result.RL[3].state

```

6.3 Generic MQTT thing code

UID: mqtt:topic:e202ba2650

label: Generic MQTT Thing

thingTypeUID: mqtt:topic

configuration:

 payloadNotAvailable: offline

 payloadAvailable: online

 transformationPattern: JS:mqtt_availability.js

 availabilityTopic: mac/0076XXXXXXXX/status

bridgeUID: mqtt:broker:8496c99244

channels:

- id: my98XX_mqtt_switch1

 channelTypeUID: mqtt:switch

 label: my9850 mqtt switch1

 description: ""

 configuration:

 qos: 0

 transformationPatternOut: MAP:mqtt_cmd.map

 commandTopic: mac/0076XXXXXXXX/cmd

 stateTopic: mac/0076XXXXXXXX/cmd_ack/id_my_openhab

 transformationPattern: JINJA:"{% if (value_json.cmd == 'getpower' and value_json.result.RL[0].state == 1) or (value_json.tag == 'r1_on' and value_json.result.message == 'success') %} 'p1_1' {% elif (value_json.cmd == 'getpower' and value_json.result.RL[0].state == 0) or (value_json.tag == 'r1_off' and value_json.result.message == 'success') %} 'p1_0' {% else %} 'p1_0' {% endif %}"

 off: p1_0

 on: p1_1

- id: my98XX_mqtt_switch2

 channelTypeUID: mqtt:switch

 label: my98XX mqtt switch2

 description: ""

 configuration:

 qos: 0

 transformationPatternOut: MAP:mqtt_cmd.map

 commandTopic: mac/0076XXXXXXXX/cmd

 stateTopic: mac/0076XXXXXXXX/cmd_ack/id_my_openhab

 transformationPattern: JINJA:"{% if (value_json.cmd == 'getpower' and value_json.result.RL[1].state == 1) or (value_json.tag == 'r2_on' and value_json.result.message == 'success') %} 'p2_1' {% elif (value_json.cmd == 'getpower' and value_json.result.RL[1].state == 0) or (value_json.tag == 'r2_off' and value_json.result.message == 'success') %} 'p2_0' {% else %} 'p2_0' {% endif %}"

```

    off: p1_0
    on: p1_1
- id: my98XX_mqtt_switch3
  channelTypeUID: mqtt:switch
  label: my98XX mqtt switch3
  description: ""
  configuration:
    qos: 0
    transformationPatternOut: MAP:mqtt_cmd.map
    commandTopic: mac/0076XXXXXXXXX /cmd
    stateTopic: mac/0076XXXXXXXXX /cmd_ack/id_my_openhab
    transformationPattern: JINJA:"{% if (value_json.cmd == 'getpower' and
      value_json.result.RL[1].state == 1) or (value_json.tag == 'r3_on' and
      value_json.result.message == 'success') %} 'p3_1' {% elif
      (value_json.cmd == 'getpower' and value_json.result.RL[1].state == 0) or
      (value_json.tag == 'r3_off' and value_json.result.message ==
      'success')%} 'p3_0' {% else %} 'p3_0' {% endif %}"
    off: p1_0
    on: p1_1
- id: my98XX_mqtt_switch4
  channelTypeUID: mqtt:switch
  label: my98XX mqtt switch4
  description: ""
  configuration:
    qos: 0
    transformationPatternOut: MAP:mqtt_cmd.map
    commandTopic: mac/0076XXXXXXXXX /cmd
    stateTopic: mac/0076XXXXXXXXX /cmd_ack/id_my_openhab
    transformationPattern: JINJA:"{% if (value_json.cmd == 'getpower' and
      value_json.result.RL[1].state == 1) or (value_json.tag == 'r4_on' and
      value_json.result.message == 'success') %} 'p4_1' {% elif
      (value_json.cmd == 'getpower' and value_json.result.RL[1].state == 0) or
      (value_json.tag == 'r4_off' and value_json.result.message ==
      'success')%} 'p4_0' {% else %} 'p4_0' {% endif %}"
    off: p1_0
    on: p1_1

```

6.4 Attribute Settings: (e.g., Current, Voltage, Temperature)

Enter in State Transformation on P.33:

***Notice:** Attributes for each relay may not be consistent.
Do not directly copy the same code for configuration.

(1) Temperature: JINJA:{{value_json.result.global_measure.temperature}}
Voltage: JINJA:{{value_json.result.global_measure.voltage}}

(2) Current: JINJA:{{value_json.result.RL[0].current}}

Disclaimer:

The content of this manual is provided for reference and general informational purposes only for IP Power users, and may be subject to change without prior notice. AVIOSYS International, Inc. make no express or implied warranties or representations regarding the accuracy, completeness, and reliability of the information on this manual related to IP Power applications with openHAB.

Any information obtained from this manual is at the user's own risk. We disclaim any and all liability for any direct or indirect losses or damages, including but not limited to loss of profits, business interruption, data loss, or other financial losses, even if we have been advised of the possibility of such damages.

This manual may contain links or references to third-party websites. We assume no responsibility for the content or availability of these third-party websites and do not endorse, guarantee, or make any representations about them. You should assess and assume the risks of using these third-party websites on your own.

We reserve the right to change or modify this disclaimer at any time. It is recommended that you regularly check this page for any updates. By continuing to use this manual, you agree to comply with these terms and conditions.

If you have any questions regarding this disclaimer, please contact us.