

# Hubitat MQTT application beta 1

IN PROGRESS

This has been 'mostly' updated now for the beta release - please let me know any issues or suggestions.

Later / release version of this read me will contain appropriate screenshots rather than just being text based.

Pre Release notes for beta 1  
25th January 2020

## # MQTT

MQTT full featured client for Hubitat Elevation Hubs (HE).

The licence for this beta has been updated to be restrictive whilst in testing. I intend the final version to be under a much less restrictive licence although it will likely require my express written permission to publish any derivate code and no commercial offering will be allowed.

Usage of this code applies full acceptance of the licence and failing technical / legal enforceability the spirit in which the licence was intended.

You may not redistribute this code in any form to any other person. It is provided to members of a restricted group for the purpose of testing and feedback only. You may use and modify this code for your own personal use only.

This read me is now mostly updated for the beta release but is still a 'work in progress'. Before installing please backup first though and as always - use entirely at your own risk and discretion, there is no warranty or accepted liability as to this app being suitable for any purpose whatsoever and no liability for any issues or damage caused by this app in whatsoever form may be claimed against the author.

---

## # Installation Notes

You must install two components - the MQTT app and the MQTT client driver. The previous (alpha4) two drivers for 'MQTT switch' and 'MQTT dimmer' are no longer required and are deprecated / will no longer function. Instead you can now use any of the 24 Hubitat virtual drivers. The MQTT text driver is still available (optional) but is likely not required nor has it been tested with beta.

If you have a pre alpha 5 version, you may lose/have to recreate any virtual devices that you have that import devices from MQTT into HE. If this is a concern contact me first as there is a way around it. You might wish to retain previous devices using my now deprecated drivers for the

purpose of copying the MQTT details over to the newer HE Virtual drivers.  
Warning: Purging MQTT devices will delete these older devices permanently.

This read me will be updated as documentation evolves but will be replaced by a richer format supporting screenshots. There are three discussion groups for this beta. A general topic and then one for HomeAssistant issues and one for OpenHAB issues.

<https://community.hubitat.com/t/beta-mqtt-app/32750>

<https://community.hubitat.com/t/beta-mqtt-app-home-assistant-discovery-and-statestream-support/32761>

<https://community.hubitat.com/t/beta-mqtt-app-homie3-protocol-support-openhab-athom-homey-etc/32762>

If you do find bugs then this GitHub repository is the place to post them under 'issues'. I notice them quicker and can create a discussion on them individually.

<https://github.com/xAPPO/MQTT/issues>

## # Features

a) Enabling inbuilt HE devices\* to publish and be controllable through MQTT either using a compliant homie3 topic or alternatively a simplified minimal topic structure. The simplified topic is the replacement for the previous alpha's 'Hubitat' topic structure. It removes most static topics from the homie3 tree and publishes mainly state changes, many of which can be updated by publishing to the state topic but with '/set' appended to the topic. In this minimal form the topic is no longer compliant with the homie3 or homie4 specification.

b) Enables automatic discovery and selected inclusion and control of HE devices\* and sensors into other controllers supporting the homie3 protocol (promoted by openHAB).

c) Enables automatic discovery and selected inclusion and control of HE devices\* and sensors into other controllers using the Home Assistant discovery protocol (promoted by HA)

d) Enable automatic discovery and selected inclusion and control of external devices\* and sensors into HE that are using the homie3 protocol.

e) Enables automatic discovery and selected inclusion of external devices\* and sensors into HE that are using the Home Assistant statestream protocol. HA statestream does not support 'control' so small automation scripts are provided for HA that add this control capability back into HA too. This supports switches and lights and can be extended to other devices by adding additional automation scripts to HA.

f) Enables existing MQTT devices\* to be 'mirrored' as any of the HE virtual devices and controlled within HE. All the topics are configurable as are the state (attribute) values. Later referred to as 'manual' or 'ad hoc' devices.

Athom's Homey controller also supports MQTT and the homie3 protocol. They work really well together automatically discovering each other's devices and enabling real time synchronisation and control between the controllers. Two HE's could be setup this way too but HubConnect is the recommended option. HubConnect virtual devices of course work with this MQTT app too.

#I have not yet tested the just released openHAB 2.5 with homie3 discovery but it may still need work. If you do use OH 2.5 (earlier versions will not work) then let me know how it goes. I have since been advised it works well.

\*N.B. This version currently supports 'switch' (onoff), 'switchLevel' (dim) colour, most sensors and a lot of other device types too - and will be expanded to include others as needed (requested).

## # Future Features:

- 1) Support of many more device capabilities.
- 2) Support Home Assistant MQTT discovery protocol bidirectionally - HE devices are already auto discovered by HA but support devices advertising using this protocol for discovery by HE. (Update: I may not implement this latter support for HA Discovery > HE as there are virtually no such devices and HA doesn't support exporting it's devices this way either)
- 3) Support for multiple homie discovered devices.
- 4) Support for JSON payloads
- 5) Support multiple MQTT brokers (considering)
- 6) Improved auto matching of device capabilities between different controllers

# # Instructions:

(preliminary)

## # Initial setup:

1) Install both the main app MQTT and the device driver for MQTT Client and optionally MQTT text.

2) From 'Devices' "Add Virtual Device" selecting the "MQTT client" device driver, name the broker device as you wish, make up a Device Network ID e.g. the IP of the broker. Important: Do NOT create two MQTT client drivers as this causes known issues.

3) Configure this device to access your existing MQTT broker using the 'Preferences' section on the next screen in the device driver then click 'Done'.

### Preferences

|  |  |  |
|--|--|--|
| <b>MQTT Broker Address *</b><br>e.g.<br>tcp://192.168.1.17:1883<br>tcp://192.168.1.78:1883 | <b>MQTT Username</b><br>(blank if none)<br>kevin | <b>MQTT Password</b><br>(blank if none)<br>..... |
|--|--|--|

Save Preferences

### Device Information

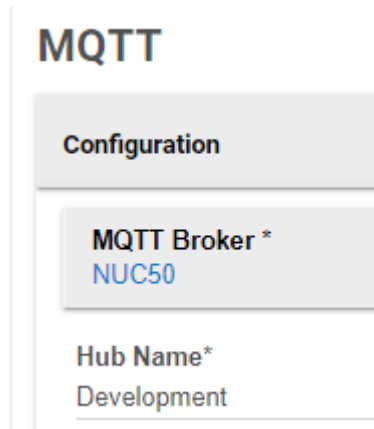
|                               |   |
|-------------------------------|---|
| <b>Device Name *</b><br>NUC50 | <b>Device Network Id *</b><br>0eeef2f0-6fa5-4878-aeed-5e062 |
| <b>Device Label</b>           | <b>Type *</b><br>MQTT Client                                |
| <b>Zigbee Id</b>              |   |

Save Device

You must enter the IP (or URL) e.g. tcp://192.168.1.78:1883 , username / password are only required if your broker needs them. NB include tcp:// at the start of the entry.

4) Launch the MQTT app, it has three setup pages each with multiple sections, the first page covers individual devices and the second and third pages are optional. The second page is for the 'manual' importing of existing MQTT devices into HE and the third for automatic discovery of devices into HE using either homie or HA statestream 'discovery'. Configuring the second and third pages is optional.

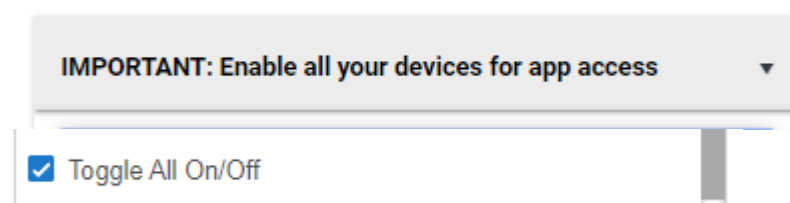
On the first page click 'Configuration' and from the 'MQTT Broker' dropdown select the MQTT client device you just configured in step 3



Give you hub a name in 'Hub Name'

Next an important step that is required for speed:

The MQTT app can only interact with HE devices that you give it permission to use. To make it possible for the app to know and match the names of all your devices on the first page click 'Configuration' and then at the bottom click this button "IMPORTANT: Enable all your devices for app access<" and enable all devices by clicking the top option. Please as you use the MQTT continue to check that all devices remain enabled - especially as you add new devices. This speeds up the app considerably rather than the fallback of having to loop through every device individually.



For the time being ignore all other options and select 'Next' on the first and then second page and 'Done' on the third.

.. you should see something similar to this in the log.

Log:

```
info MQTT: ===== Startup complete =====
info MQTT:      0 Hubitat devices enabled on MQTT
info MQTT: =====
info MQTT: Skipping HA stateStream MQTT discovery
info MQTT: Skipping homie MQTT discovery
info MQTT> Connected as Hubitat_Development to MQTT broker
tcp://192.168.1.78:1883
```

```
info MQTT client alpha 5 initialised
info MQTT> Resetting MQTT connection
info MQTT> Log Level set to 2
info MQTT: Hubitat hub name is : Hubitat/Development
info MQTT alpha 5 Initialized
info MQTT: MQTT Installed
```

In particular check that you get the "Connected as Hubitat\_xxxx to MQTT broker" success result and no other errors, otherwise nothing will work !

Now choose from the key features listed above which feature(s) a) b) c) d) e) or f) you are wishing to setup the MQTT app for.

I recommend choosing just one for now preferably a) as it is the easiest.

Now follow the matching option a)-f) below that you need.....

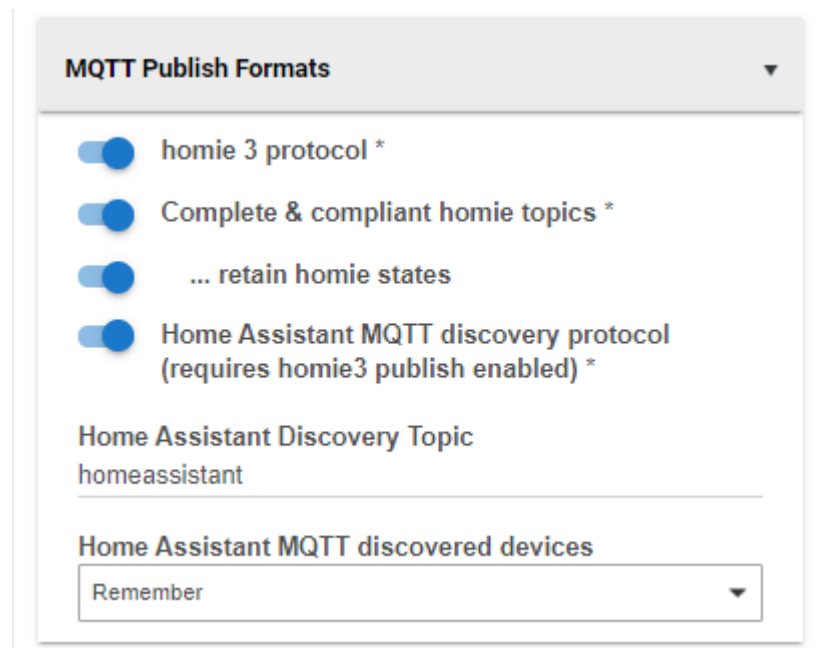
Lastly several of the less obvious app configuration options are discussed right at the end of these instructions in 'Additional Notes'. Please read those too.

---

# # a) Enabling inbuilt existing HE devices\* to publish and be controllable through MQTT

This is used to expose selected existing HE devices onto MQTT so they report their status and can be controlled via MQTT

From 'Apps' run the MQTT app again and select how you would like to use to publish your devices to MQTT.



The screenshot shows the 'MQTT Publish Formats' configuration interface. It features four toggle switches, all of which are turned on. The first toggle is labeled 'homie 3 protocol \*'. The second is 'Complete & compliant homie topics \*'. The third is '... retain homie states'. The fourth is 'Home Assistant MQTT discovery protocol (requires homie3 publish enabled) \*'. Below these toggles, there is a section for 'Home Assistant Discovery Topic' with a text input field containing 'homeassistant'. At the bottom, there is a section for 'Home Assistant MQTT discovered devices' with a dropdown menu currently set to 'Remember'.

You need to enable the 'homie3 protocol' option via the MQTT Publish formats

You have two choices - the standard "homie3 protocol" compatible format or a reduced (simplified) version of this.

The previous alpha4 'Hubitat Basic' topic structure is now deprecated and replaced by the simplified option above.

homie3: and simplified topic

```
homie/hubitat_hubname/deviceName/capability/<value>
homie/hubitat_hubname/deviceName/capability/set/<value>
```

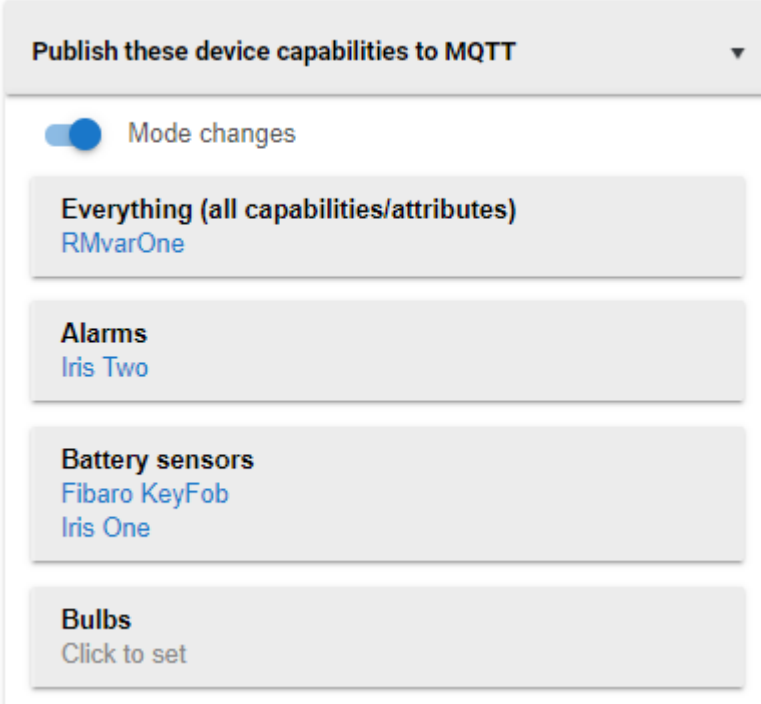
NB. - To change (command) the state of a HE device (if permitted) use the state topic with '/set' appended at the end of the topic, as shown above.

homie3: full version

includes additional configuration topics required for homie3 compliance

When enabling the homie protocol there is an additional '...retain homie states' option. Enabling this causes the last reported state of your devices to be retained on your MQTT broker. Any new device clients connecting to MQTT will receive these states , although these may be old (residual) values. If you disable it any new device connecting to MQTT will get no information until your device updates its state, but the value will be known to be current. This should normally be left enabled.

Now enable which of your HE devices that you wish to publish to MQTT by selecting them in the provided dropdowns. A dropdown is provided for each capability\*\*. If you have no devices of that capability on HE then the dropdown will not be shown



**Publish these device capabilities to MQTT** ▼

☒ Mode changes

**Everything (all capabilities/attributes)**  
RMvarOne

**Alarms**  
Iris Two

**Battery sensors**  
Fibaro KeyFob  
Iris One

**Bulbs**  
Click to set

click 'Update' then 'Next' and 'Next' and then 'Done'

These devices will now update status changes to MQTT and be controllable from MQTT, using .../set appended topic.

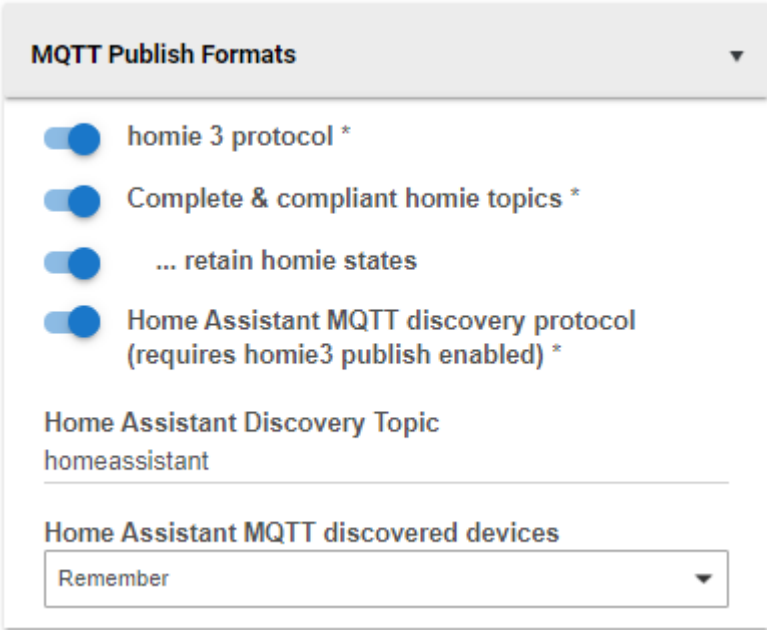
\*\* A quick explanation on 'capabilities': When you select a device in a capability dropdown that device will publish just that drop downs capabilities to MQTT - not all the capabilities it might have. So only some of the devices 'capabilities' might be reported to MQTT. Therefore make sure all the capabilities that you wish are enabled for that device, by selecting it in several drop downs, or to select every capability a device has use the 'Everything (all capabilities/attributes)' drop down at the top and select your device. For example some motion sensors provide motion, temperature, battery and light sensors. That's 4 capabilities, each individually selectable.

---



## # b) Enable automatic discovery of HE devices by other controllers (e.g OpenHAB) using homie3 protocol

In Configuration >> MQTT Publish Formats.



**MQTT Publish Formats**

- ☒ homie 3 protocol \*
- ☒ Complete & compliant homie topics \*
- ☒ ... retain homie states
- ☒ Home Assistant MQTT discovery protocol (requires homie3 publish enabled) \*

Home Assistant Discovery Topic  
homeassistant

Home Assistant MQTT discovered devices  
Remember

Enable "homie 3 protocol"

Ensure "Complete and Compliant homietopics" is enabled

Enable "...retain homie states"

Enable the HE devices you wish to be discoverable using homie by following a) above.

Configure your other controllers 'discovery option to point at HE's device homie topic.

homie/hubname/#

---

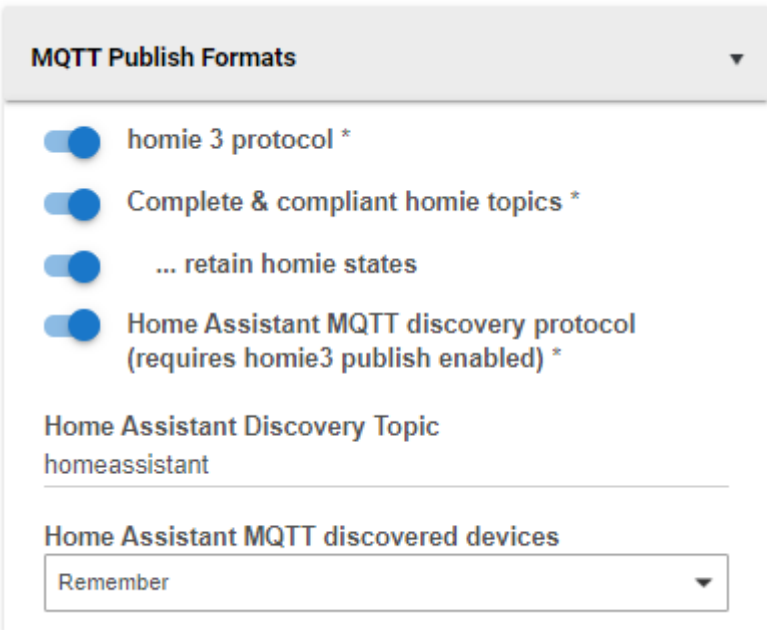
## # c) Enable automatic discovery of HE devices by other controllers (e.g. HA) using Home Assistant Discovery protocol

First enable HA MQTT discovery on your other controller.  
If using Home Assistant ensure it has 'discovery' enabled and note the topic that it is 'watching' (within the mqtt section of HA's configuration.yaml file). You will need to restart HA if you edit configuration.yaml

```
discovery: true
discovery_prefix: homeassistant
```

Do NOT use the same topic name (base\_topic) that HA is publishing statestream to - see e) below.

In Configuration >> MQTT Publish Formats.



Enable "homie 3 protocol" \*\*\*

Enable "Home Assistant MQTT discovery protocol" (requires homie3 publish enabled)

\*\*\* You must enable this as HA reads the MQTT payloads from these homie topics

In the HA Discovery Topic field enter the discovery prefix value that HA is using (e.g. homeassistant if as above).

Click through 'Next' 'Next' 'Done'

Devices that are enabled for MQTT within HE will automatically appear as entities in your Home Assistant front end. You may to have press ctrl F5

(or sometimes restart HA) to refresh the HA interface. Alternatively, devices may initially appear as 'unused entities' which you can check by clicking the 'hamburger' icon in the top right hand of recent HA builds. These devices are bi-directionally real time synched with HE.

When HA starts up it checks MQTT for discovered devices. If you want them to persist over HA restarts, then ensure you set "Home Assistant MQTT discovered devices" to 'Remember'" in the MQTT app. If it is unchecked then HA will purge these devices when restarted, although it might warn you with yellow boxes the first time it notices they have gone.

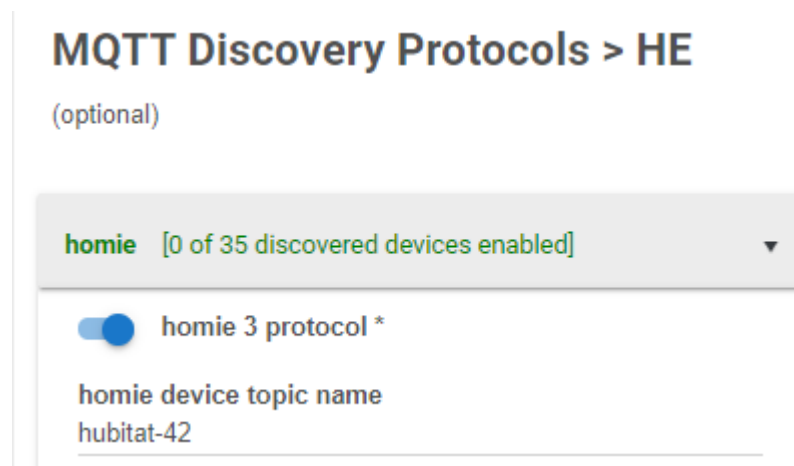
When the MQTT app has fully started it sets the \$status = ready topic. HA monitors this and at this stage HA MQTT devices will change from 'unavailable' to being functional. Should the MQTT app not complete startup (\$status = init) or the MQTT broker topics not be available to HA then related devices will show 'unavailable'

---

## # d) Enable automatic discovery and selected inclusion of devices\* using the homie3 protocol (promoted by openHAB)

On the third page of the app entitled

"MQTT Discovery Protocols > HE"



**MQTT Discovery Protocols > HE**

(optional)

**homie** [0 of 35 discovered devices enabled]

☒ **homie 3 protocol \***

**homie device topic name**

hubitat-42

select 'homie' and enable the homie 3 protocol

Enter the top level name of the homie 'device' you wish to use in 'Homie Device Topic Name'

e.g. for homie/myDevice enter 'myDevice'

All the nodes within that device will be scanned.

click 'Done'

If a homie topic looks like homie/mainsys/hallway/onoff then you would enter mainsys as 'device topic name' as that is the device identifier (mainsys probably has many other nodes like hallway)

Discovery may take a minute or so depending on how many nodes there are - the log will show you when it is complete.

Next time you enter page three the 'homie' button will be green and show how many devices were enabled / discovered and dropdowns will be present that allows you to choose which of these 'homie' devices should be imported into HE.

Discovered 0 homie switches [^0]

Click to set ▼

Discovered 3 homie dimmers [^0]

Click to set ▼

Discovered 10 homie sensors [^0]

Click to set ▼

Discovered 1 homie buttons [^0]

Click to set ▼

☐ fibaro-keyfob

Once selected and 'Done' the app will create child devices for those that are enabled, attempting to choose an appropriate virtual device type. You can manually alter the device type later in HE's Devices menu if needed.

NB You can only import one other homie device and that device must be within your existing homie topic tree (but obviously not within your HE devices homie tree)

---

## # e) Enable automatic discovery and selected inclusion of devices\* using the Home Assistant statestream protocol

Firstly configure HA to use MQTT statestream by including the following in your HA configuration.yaml (choose your own base\_topic: name)

```
mqtt_statestream:

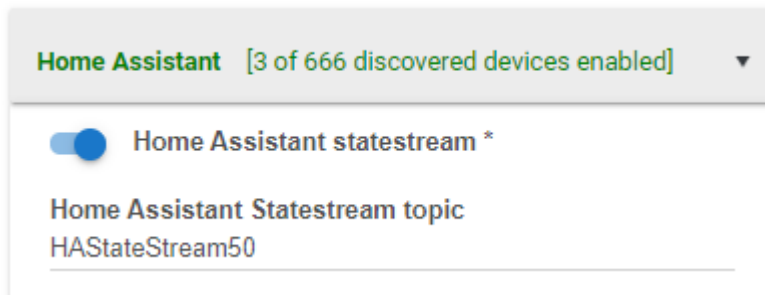
  base_topic: HAStateStream50      # << your choice of topic name

  publish_attributes: true
```

Add the included "automations.yaml" to your automations.yaml file in HA , or if you don't use this then to configuration.yaml. This enables bi-directional control via statestream (which is normally a status only capable topic).

On the third page of the app entitled

"MQTT Discovery Protocols > HE"



"Home Assistant statestream" enable this

"Home Assistant Statestream Topic" enter the topic name that your HA software is publishing to e.g HAStateStream50

Do NOT use the same topic name (discovery\_topic) that HA is itself discovering devices from (normally 'homeassistant') - see C) above

Click 'Done'

The app will then start up which may take a minute or so and automatically discover all the devices you have asked it too.

The log will inform you when Home Assistant Discovery has completed.

After completion the app will then populate the discovered device dropdown lists so you can now select which devices you are interested in.

Discovered 149 Home Assistant switches [^0]

Click to set

Discovered 108 Home Assistant lights [^1]

davenport

☐ back\_landing

☐ back\_landing\_2

☐ back\_stairs

☐ blinds\_kh

This you can see by revisiting the third page of the app where 'Home Assistant' will now be green with a count of how many devices were found.

Clicking 'Home Assistant' reveals all the drop down selectors.

Once enabled and 'Done' the app will create child devices for those that are enabled, attempting to choose an appropriate virtual device type. You can manually alter the device type later in HE's Devices menu if needed.

---

## # f) Enable MQTT devices\* to be 'mirrored' as virtual devices and controlled within HE.

Also known as 'manual or 'adhoc' devices this is the most involved option and requires you to 'manually' create virtual MQTT child devices, using any of HE's 24 inbuilt virtual drivers. Then you configure that device to map to the MQTT topics that represent its state and control. This is all done on the second page of the app entitled "Virtual MQTT Data"

### Virtual MQTT Data

(optional)

Device Type

Virtual Dimmer

MQTT enabled Virtual Dimmer devices

[vDim manual](#)

[Edit this Virtual Dimmer device](#)

... or create a new virtual Virtual Dimmer device called ...

[Click to set](#)

Next

HE can't (despite request !) create a device list dropdown for all 'Virtual Devices' so on this page you must first choose a 'Device Type' at the top and then you separately manage devices within each type category using a second 'MQTT enabled xxx devices' dropdown.

Listed in the drop downs will be all the child devices using virtual drivers in your MQTT app. This will therefore include child devices that are auto created by discovery. This allows you to manually edit those devices too here if needed.

Data for the MQTT linking between the virtual device and HE is held in .. well 'Data' within each device. You can view this data from the HE 'Devices' menu, but you can't create or edit if from there.

Below is a screenshot of this data for a dimmer device 'myNewDevice' that you will create later below.



Data

- level\_Cmd: sentinel/kitchen/spot-aga/dim/set
- switch\_OFF: off
- max\_Level: 100
- mqtt: enabled
- origin: user
- switch\_ON: on
- level\_Topic: sentinel/kitchen/spot-aga/dim
- switch\_Topic: sentinel/kitchen/spot-aga/onoff
- switch\_Cmd: sentinel/kitchen/spot-aga/onoff/set

So, this third page provides an editor for existing values and allows you to create new devices. You can't (and shouldn't need to) create new data values or completely delete old ones.

First of all, let's look at creating a new device that is going to 'mirror' an existing device on MQTT i.e. we'll import an adhoc device from MQTT into HE.

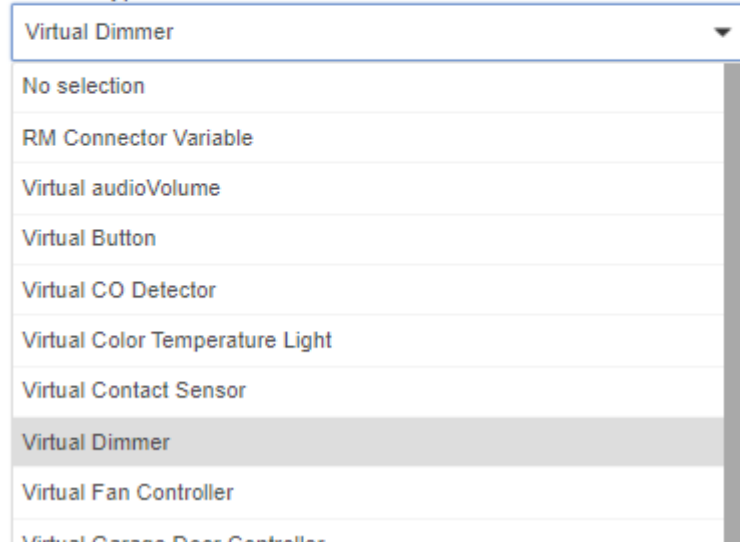
NB. This device on MQTT must not sit within the homie tree - if it does then auto discovery should be used instead. The device would not be visible as an 'adhoc' device as anything within the homie tree is ignored.

Decide what type of new device this will be in HE. You can choose from any of the 24 HE Virtual drivers (at the top of the page).

## Virtual MQTT Data

(optional)

Device Type



|                                 |
|---------------------------------|
| Virtual Dimmer                  |
| No selection                    |
| RM Connector Variable           |
| Virtual audioVolume             |
| Virtual Button                  |
| Virtual CO Detector             |
| Virtual Color Temperature Light |
| Virtual Contact Sensor          |
| Virtual Dimmer                  |
| Virtual Fan Controller          |
| Virtual Garage Door Controller  |

Choose one and then click 'Edit this Virtual XYZ device"... make sure NO devices are selected in this dropdown (by deselecting) and hit return or click outside of the drop down. Selecting no device here allows a new device to be created, selecting an existing device edits that existing device.

Edit this Virtual Dimmer device

☐ vDim manual

Update

... or create a new virtual Virtual Dimmer device called ...  
Click to set

Create device Test Device

NB: The pages are dynamically created (they change content based on selection) and this is proving a little slow on HE - please allow a couple of seconds for the pages to redraw when you edit things on this page. I will work on speeding this up but it's outside of my control really.

In the text entry box type a unique name e.g. 'myNewDevice' for your new device and hit return - then click 'Create device myNewDevice'.

Create device myNewDevice

Now, after the page refresh select the device and enable it in the drop down 'MQTT enable Virtual XYZ devices'.

MQTT enabled Virtual Dimmer devices

vDim manual

☒ myNewDevice

☒ vDim manual

For 'adhoc' devices to be useable via MQTT they **must** be enabled in this drop down. This does not apply to Discovered devices - they are enabled on the third page of the app. However, to **edit** an adhoc device it **must** be enabled on this page too, otherwise it displays no data values. This is a safeguard as 'discovery' auto completes what it deduces are the are correct values for data.

So, let's link this new device 'myNewDevice' to MQTT.

After enabling the device select 'myNewDevice' in the lower dropdown 'Edit this Virtual XYZ Device' and click 'Update'. This will then populate the bottom of the screen with settings that contains the MQTT topics (and some default state values) for this new device. We will next edit those to match the devices actual values on MQTT. There is also a button to delete the device should you decide you no longer need it.

Edit this Virtual Dimmer device  
[myNewDevice](#)

Delete device myNewDevice

Method to add device attribute topics \*

Manual

level attribute MQTT status topic

level MQTT command topic

switch attribute MQTT status topic

switch MQTT command topic

Please enter the FALSE status value for **switch** attribute  
off

Please enter the TRUE status value for **switch** attribute  
on

Please enter the maximum possible level:  
100

Please leave "Method to add device attribute topics" set to manual for the time being.

Depending on the virtual devices 'attributes' a different number of values are displayed. This allows you to individually map device attributes to topics on MQTT. If the attribute is 'settable' then both an

"[attribute] MQTT status" topic and

"[attribute] MQTT command" topic"

value are shown, if it is not then only

"[attribute] MQTT status topic"

will be offered.

"[attribute] MQTT status topic" contain the MQTT topic address for the state payload for this attribute (HE reads from these)

"[attribute] MQTT command topic" values contain the topic address for the cmd payload for this attribute to request a state change (HE writes to these)

e.g.

"temperature Status value" is a sensor input and might have a 'topic' data value of

**myDevice/BedroomMain/temperature1** (containing a payload = 22.6)  
it would not have a Command data value as it can't be controlled

"switch MQTT status topic" might have a 'topic' value of  
**myDevice/BedroomMain/heater** (containing a payload = OFF)

"switch MQTT command topic" might have a 'topic' value of  
**myDevice/BedroomMain/heater/control** (and might expect a payload of 'OFF' or 'ON' )

'switch' and 'temperature' could both be HE attributes of the one device or be within separate devices.

(optional)

"Please enter the ON status value for switch" is the expected MQTT value for the boolean 'ON' condition e.g. On or High

"Please enter the OFF status value for switch" is the expected MQTT value for the boolean 'OFF' condition e.g. false or 0

These may differ from HE's expected boolean values which for a contact sensor for example is 'open' 'closed'. You can leave these blank to pass a value 'as is' or you can map your own values to HE's e.g. 'Ein' and 'Aus' to 'on' and 'off' if needed

So, here's some typical data for a contact sensor as shown in the 'Devices' menu from HE

```
contact_Topic: SHController/state
contact_OFF: OFF
mqtt: enabled
contact_ON: ON
contact_Cmd:
```

or a dimmer device as displayed on this edit page

level attribute MQTT status topic  
sentinel/kitchen/spot-aga/dim

---

level MQTT command topic  
sentinel/kitchen/spot-aga/dim/set

---

switch attribute MQTT status topic  
sentinel/kitchen/spot-aga/onoff

---

switch MQTT command topic  
sentinel/kitchen/spot-aga/onoff/set

---

Please enter the FALSE status value for **switch** attribute  
off

---

Please enter the TRUE status value for **switch** attribute  
on

---

Please enter the maximum possible level:  
100

---

If you look back at the Data for 'myNewDevice' (4 pages back) you will see that it reflects these entries - for example the switch\_Cmd value is equivalent to the value you entered here

switch MQTT command topic

Of special note for level based devices is this  
max\_Level: 100  
.. or in the app  
"Please enter the maximum possible level" 100

This allows you to seamlessly map a device on MQTT that has levels of say 0-255 to an HE device all of which use levels 0-100. This works for both status and command messages. Athom's Homey uses values between 0 and 1 so you would put 1.0 in here

Now that might seem a lot of information, but it really is very easy and yet very flexible. Work through it and you'll get the hang very quickly, typically it's only one or two topic values you'll need to enter. More complex virtuals like Virtual Multi and especially my 'most hated' Virtual Omni are more challenging though as they have lots of attributes.

After configuring click through 'Done' and the new subscriptions will be added for your new device. The current values for the state(s) will be updated and control should be available from HE

---

# # Additional Notes:

After successful configuration and Startup you should see something like the following in the log at 'INFO' level. Individual device types are also broken down.

```
info MQTT: ===== Startup complete =====
info MQTT:      Discovered 70 HA light devices
info MQTT:      Discovered 47 HA switch devices
info MQTT:      Discovered 13 homie dim devices
info MQTT:      Discovered 10 homie onoff devices
info MQTT:      23 Hubitat devices enabled on MQTT
info MQTT: =====
```

There are a few other app configuration options to be aware of.

## # LogLevel:

LogLevel (within Configuration) can be set to limit the logging entries the app posts. It is recommended to set it at level 'INFO' but it can be minimised to "WARN". Any log messages that show in red as 'ERROR' need investigation (and reporting). Warn message are normally something just to be aware of.

## # Purge Discovered Devices

WARNING: Setting this will delete all MQTT 'discovered devices' when you click 'Done'

Switching this ON will clear all the apps child HE devices that were added through 'discovery' but not manual 'ad-hoc' configuration.

Discovered devices can be recreated by re-running the app. However, the unique internal device ID for the device will then change and so the newly created devices will be just that ... 'new'.

This really only has an impact on HE internal references to the original device e.g. Dashboard devices and RuleMachine rules. You will have to recreate these.

The main use is to clear out all your discovered devices but once running happily you should leave this set to OFF.

You can always disable devices on the second page of the app config and individually delete previously created devices manually.

## # Forget Enabled Devices

Enabling this will purge your drop down lists from devices that have been discovered and enabled in these lists at restart. They will be re-discovered at startup but they will not be re-enabled.

## # Complete and Compliant Homie Topics

This option publishes a minimal but non 'homie3 / homie4' compliant homie/tree . It is simpler essentially by only presenting state values for device attributes that also (if settable) can be updated by appending /set to the end of the topic and publishing an 'allowable' payload. This is useful if you have no interest in the homie protocol and just wish to use HE/MQTT in a simple form. This likely works for HA Discovery but will obviously not

work for homie discovery. The 'homie' top level is still mandatory as that allows you later to switch over to a full homie3 usage without having to rename all your topic links. It will not be configurable or omissible.

### **# Retain homie states**

Makes the homie state topics retained on the MQTT broker (normal - as per spec). NB homie /set topics should NOT be published retained.