Heating X2: Schedule Thermostats with Calendars

Code Generator Instructions

Andy Symons

3 February 2024

Introduction

The **Heating X2 code generator** generates all the YAML code you need to define the helpers and timers required by the Heating X Blueprints for an entire home installation. There are additional YAML code generators for optional thermostat template sensors, the automations, and example dashboard cards.

The code generator uses Microsoft Mail Merge, so you need a system running Microsoft Office (sorry, I could not find a free app for this).

Preparing your system

The code generator is going to generate code to paste into separate files for each type of helper, the timers, and the templates, so you need to have these files available if they are not already. Separating the files is good practice for larger systems anyway, where putting everything in the configuration.yaml file would be unwieldy.

1) Make a full backup
2) Using a code editor such as Home Assistant File Editor or Studio Code Server, create the following empty files in the /CONFIG directory, unless you already have them:
   - input_datetimes.yaml
   - input_numbers.yaml
   - input_texts.yaml
   - templates.yaml
   - timers.yaml
3) If you already have some input datetimes, input numbers, input texts, templates, or timers defined in configuration.yaml, move them to the new files you just created.
4) In configuration.yaml enter the following text unless you already have it:

```yaml
input_datetime: !include input_datetimes.yaml
input_number: !include input_numbers.yaml
input_text: !include input_texts.yaml
template: !include templates.yaml
timer: !include timers.yaml
```

Download the code generator files

Download the 'Heating X2 Code Generator.zip' file and unzip it into a folder somewhere on a *local* drive such as 'downloads'. Mail Merge has problems if run on a cloud drive such as iCloud or OneDrive. The files are temporary; you only need them long enough to set up your system; they are not *part of* the system; Home Assistant will keep the YAML files that you create.

After unzipping, you will have a folder with an EXCEL workbook for your Room and Thermostat data, and a number of word files corresponding to the YAML files to be created, numbered in the sequence we are going to use them here:

- 1 Heating X2 code generator DATA.xlsx
- 2 Heating X2 code generator for input_numbers.yaml.docx
- 3 Heating X2 code generator for input_texts.yaml.docx
- 4 Heating X2 code generator for input_datetime.yaml.docx
- 5 Heating X2 code generator for timers.yaml.docx
- 6 Heating X2 code generator for templates.yaml.docx
- 7 Heating X2 code generator for automations.yaml.docx
- 8 Heating X2 code generator for dashboard cards.docx

Enter your Room and thermostat information

Open ‘Heating X2 code generator DATA.xlsx’. This is data source for all the later mail merges. It initially contains details of a fictitious installation by way of example. You should delete the data in the yellow cells and replace it with your own. Add or remove lines as required, but do not touch the grey cells with the formulae in them.

There is a line for each Thermostat, grouped by Room. In column A enter the Room name in text with normal capitalisation and spacing, e.g. Dining Room, Kitchen, or Annie’s Room. Apostrophes are allowed. One Room corresponds to one Calendar (which you have to create in the UI) and one Automation. Words like ‘heating’ and ‘thermostat’ are added later, so do not include them in the Room name.

Columns B and C are used when there are multiple thermostats for the same Room (to be run from the same Calendar and Room sensors). Repeat the Room name in column A, number the thermostats in column B and give them distinct names for their location within the Room in column C – such as North, South, East and West in my example. The number and name are not needed when there is only one thermostat in the Room.

Column D is for the type of thermostat to be used in the name. It will be typically ‘TRV’ or ‘Thermostat’, or perhaps ‘Generic thermostat’ if you created it in Home Assistant.

Column E is for the entity id of whatever switch or sensor you are using to determine the ‘Away’ mode for the Room. In my example it is a simple Input Boolean on the dashboard. Typically all Rooms will be linked to the same switch or sensor but you can have more than one. Leave this column blank if you do not want the Away mode facility at all, or leave individual fields blank if you want some Rooms to be unaffected by Away mode.

The blue cells give *recommended* names for your calendars and thermostat devices and are used as such in the code generator unless you change them. If you used different names when you created the calendars and thermostat devices, you can put the actual name here or change the actual name to match the recommendation.

Once this is done, we are ready for the first code generation. Save and close the workbook for now, but you can always come back if you change your mind.

Generate the input numbers

Open the file ‘Heating X2 code generator for input_datetimes.yaml.docx’ in WORD.

If asked to specify the data source, select (your edited copy of) ‘Heating X2 code generator DATA.xlsx’. Agree to the security warning, and accept the default range of ‘entire worksheet’.

Select the ‘Mailings’ tab in WORD. In case the file has opened *without* asking for a source, click ‘Select recipients’ … ‘Use an existing list’ … and select (your edited copy of) ‘Heating X code generator DATA.xlsx’. This will ensure that your latest data is used.

Click ‘Finish and merge’ and select ‘Edit individual documents’. This opens a new WORD document (typically called ‘Letters1’, ‘Letters2’ etc.), which contains all the YAML code for input_datetimes.yaml.

You do not need to save this file, just copy the entire contents onto the clipboard and paste it into the ‘input_datetimes.yaml’ in Home Assistant. You can then close the ‘Letters’ file without saving it.

If you were a Heating X release 1 user, you should delete or overwrite the helpers from that installation.

Generate the input texts

Open the file ‘Heating X2 code generator for input_texts.yaml.docx’ in WORD.

Follow the same instructions as for input numbers.

Paste the contents of the mail merged file into the file ‘input_datetimes.yaml’ in Home Assistant, then close it without saving it.

If you were a Heating X release 1 user, you should delete or overwrite the helpers from that installation.

Input datetimes

Actually, release 2 does not need any input datetimes. This is just a reminder to release 1 users to delete any previously created Heating X helpers in this file.

Generate the timers

Open the file ‘Heating X2 code generator for timers.yaml.docx’ in WORD.

Follow the same instructions as for input numbers to create the YAML code.

Paste the contents of the mail merged file into the file ‘timers.yaml’ in Home Assistant, then close it without saving it.

If you were a Heating X release 1 user, you should delete or overwrite the helpers from that installation.

Generate the templates

The Heating X2 blueprint does not require template sensors, but the code is still available in case you want them anyway for use in dashboards or other automations. There are sensors for set and measured temperatures and a new one for heat demand.

If you already have template sensors for set and measured temperatures from Release 1, and do not need the one for heat demand, there is no need to make any change.

The new heat demand sensor is intended for zone control. If you have a heating system with Rooms divided over a number of zones through which hot water passes when an electric zone valve is open, or a boiler that needs the heating controller to tell it when heat is demanded, then you need zone control. If the zone valves or boiler start-up are connected to smart switches, then they can be controlled as follows:

1. Use the code generator to make the code for the thermostat template sensors including the third new ‘heat demand’ binary sensor. The third sensor uses the other two.
2. Create (by hand in the UI) a binary sensor group for each zone. To each group add the heat demand sensors for the thermostats in the zone. The zone group will be ‘on’ if any one of the thermostats is demanding heat, and off only if none of them are.
3. Make a simple automation (by hand in the UI) for each zone, that is triggered by a change in the heat demand group state and turns the zone control switch on or off as applicable.

If you regenerate Release 1 template sensors, you will find that duplicate entities appear in the because I previously used random numbers for the unique ids. You therefore need to delete by hand in the entities list all the versions that have the error ‘no longer provided by …’ and edit the new ones to use the correct entity name without a number 2 appended.

The good news is that I learned from this mistake and the code generator now uses fixed unique ids based on the entity id, so next time there should be no duplication.

Create the Automations

A new mail merge in release 2 generates the code for automations.yaml (a file you will already have) for each Room, invoking the blueprint with the correct thermostats, helpers, and default parameters. This saves time to get started by ensuring that the right helpers and timers are linked to the right automations, which is easy to get wrong in the UI. The new automations can and should be edited in the UI afterwards.

Create the YAML by mail merge as in the previous steps. The YAML code should be added carefully to the automations.yaml file, taking care not to touch your existing automations.

For the helpers, template sensors and automations to take effect you need to restart Home Assistant. Check the configuration first in developer tools and address any problems that are reported. Then, a ‘quick reload’ will suffice.

You should check all the automations in the UI after generation in case you have any mismatched names. Due to limitations of mail merge, the automations will only have one thermostat, so any others have to be added by hand. Add any door and window closure sensors, and any presence sensors by hand. You might want to adjust the parameters for some Rooms away from the defaults.

Create a dashboard

A dashboard is not part of the blueprint, and you can build one however you like, but if this is a new, large installation, you might like an example dashboard to get you started.

My example card per Room is a vertical stack with the name of the Room as its title, the first thermostat for that Room, the reason text as a markdown card (because the text is too long for a regular text entity) and the manual countdown timer (which give you the possibility to reset it).

I use Mushroom cards and Better Thermostat in my example, so you will need to load these first if you do not already have them.

To create the YAML, open ‘Heating X2 code generator for dashboard cards.docx’ and use mail merge as in the previous steps. The result is a YAML file with a card for each Room. You can use the generated code as a whole to create an entire view, or you can take each one separately and put them on a dashboard one at a time.

To create an entire view that will display all the Room cards (in the same order you listed them in your DATA sheet), I recommend you use the dashboard editor UI to create an empty View and name it. Then select the ‘Edit raw configuration’ option in the dashboard editor, find the empty view you just created and paste the YAML code in it.

If you prefer to add one card at a time, on any view press ‘add card’ and select the ‘manual’ card. Paste the YAML for one Room into that card.

Due to the limitations of mail merge, only the first thermostat for each Room is included in the card, but there is a place holder to add a second one (or more) where applicable within a grid two thermostats wide. Just copy the code for the first one and modify the entity name.

While testing, you might like to add some of the helpers and timers to the cards so that you can better see what is going on. Adding timers gives you the possibility to cancel them.

Good luck!