User & Technical reference manual

Contents

1 Introduction

NSPanel Manager is a custom software solution for the Sonoff NSPanel (not the NSPanel pro). The software is designed to be easy to use on a day to day basis and to easily manage multiple NSPanels around your home. The interface on the NSPanel itself has been designed to be intuitive to use for people of all ages and backgrounds.

All the NSPanel that are installed with the NSPanel Manager solution communicate back to a Docker container that is used to manage the panels, NSPanel Manager specific solutions and also all communication back and forth to/from Home Assisant and/or OpenHAB.

Info: For the latest release of this document, please check the GitHub page here.

2 Quick start guide

2.1 Container installation/update

2.1.1 Windows

Important: There are currently problems running NSPanel Manager container on WSL2 as WSL2 does not handle networking properly. For more information, see the following issue on GitHub.

If you wish to run the NSPanel Manager container on Windows the current solution is to run it in a virtual machine. When you have your virtual machine up and running, you can follow the "Standalone docker container" installation below.

2.1.2 Standalone Docker container

Installation
Prerequisites:

- Working Docker installation.

The pre-built container image is available on Docker hub as several different architectures, the following architectures are available:

- armhf
- armv7
- aarch64
- i386
- amd64

In order to run one of these images, issue the docker run command with the appropriate image for your hardware. The following is an example on how to start the image on a x64 PC with timezone Europe/Stockholm and store the data from the container in the "/nspmdata/"-directory:

```
docker run --name nspanelmanager -e TZ=Europe/Stockholm -v \
"/nspmdata/":"/data/" \
-d -p 8000:8000 nspanelmanager/nspanelmanager-amd64:latest
```

Important: All the data for the container will be stored in the directory mapped to /data/ in the container, in the example above this is the /nspmdata/-directory on your machine.

If you wish to manually build and run the container or change options or settings, see below for advanced setup.

Update an existing installation
When a new updated is released, do the following to update the standalone docker container:

Important: Create a backup of the "nspanelmanager_db.sqlite3"-file in the /nspmdata/-directory as it contains all settings and stored data for the container.

In order to update to the latest version of the container you have to remove the current image and recreate it. Perform the following in order to update:

- Run `docker rm -f nspanelmanager`.
- Run `docker rmi nspanelmanager`.
- Redo the installation steps above and make sure to point to the same data_dir as previously.

Info: Container management can be made much easier with the help of tools like portainer.

2.1.3 Home Assistant add-on

Installation

- In the Home Assistant web interface, navigate to Settings → Add-ons → Add-on store.
- In the upper right corner, press the three dots and choose "Repositories".
- Add **https://github.com/NSPManager/NSPanelManager** to the list of repositories and close the dialog.
- Select the "NSPanel Manager" add-on and install it.
- Check that the add-on should start automatically.
- Start the add-on.

Update an existing installation
When a new update is release, do the following to update the container currently installed in Home Assistant:

- In the Home Assistant web interface, navigate to Settings → Add-ons and check for updates.
- Select the "NSPanel Manager"-add-on and choose to update it.

2.2 Container settings

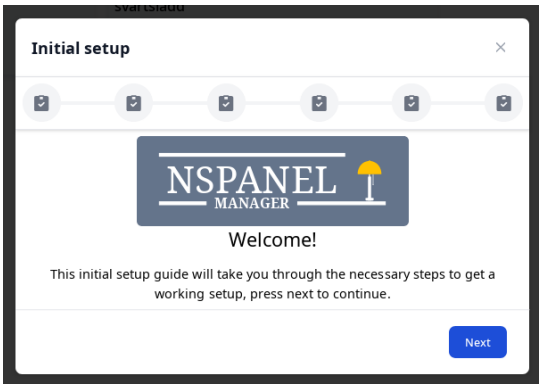


Figure 1: Initial setup

When first logging in to the NSPanel Manager web interface you will be greeted by a **Initial setup** dialog. This guide will help you setup the required items in order to have a functional setup. The following is a walkthrough of what to enter:

- **Manager config**
  - **Manager address** - This address is sent to NSPanel over MQTT when the request to connect with a manager. This can be loaded from the URL using the "Load from URL" button. This address needs to be reachable by all NSPanels.
  - **Manager port** - This port that will be used to connect to the address above.
- **MQTT config**
  - **MQTT address** - The address to your MQTT broker. If you are running your MQTT broker as an addon to Home Assistant, enter you Home Assistant address (only IP).
  - **MQTT port** - The port to connect to your MQTT broker.
  - **MQTT username** - The username to authenticate to your MQTT broker. Leave empty if you dont́ use authentication.
  - **MQTT password** - The password to authenticate to your MQTT broker. Leave empty if you dont́ use authentication.

- **Home Assistant config (optional)**
  - **Home Assistant address** - The address to your Home Assistant instance. This address should include "http" or "https" in the beginning and ":8123" (change to your port) at the end.
  - **Home Assistant token** - The long lived access token used to authenticate to Home Assistant.
- **OpenHAB (optional)**
  - **OpenHAB address** - The address to your OpenHAB instance. This address should include "http" or "https" in the beginning and ":8123" (change to your port) at the end.
  - **OpenHAB token** - The access token used to authenticate to OpenHAB.

Once the above is completed, continue on with NSPanel flashing to flash the NSPanel Manager firmware on to a NSPanel.

2.3 NSPanel flashing

Prerequisites:

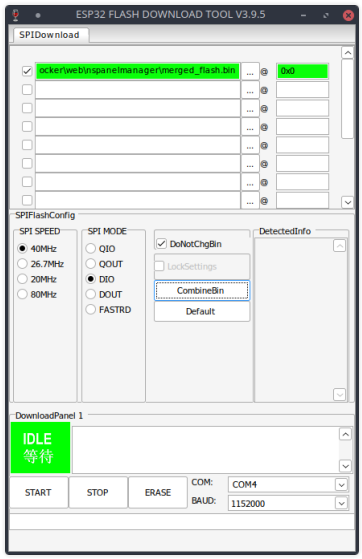
- Working TTL flasher for 3.3V.
- Working serial setup for your PC and known serial port (in Windows known as COM-port).

In order to connect to the NSPanel and be able to flash it, you must dismantle it. For a guide on how to dismantle and connect your serial flasher to the NSPAnel, refer to this guide from MarkWattTech.

2.3.1 Flashing with Espressif ESP32 DOWNLOADER TOOL (Windows only)

To flash the panel, perform the following:

- Download the tool from Espressif from here.
- Open the tool and choose to flash an ESP32 chip.
- Check one checkbox and select the "merged_flash.bin"-file in the "docker/web/nspanelmanager/"-directory.
- Enter "0x0" as the upload address.
- Connect your flasher to the NSPanel and press "START".



2.3.2 Flashing with ESPtool (all operating systems)

By installing esptool it is possible to upload the merged flash using the command line. Do the following:

- Open a terminal.
- Navigate to the "docker/web/nspanelmanager/"-directory.
- To determine if you have selected the right port, run `esptool.py flash_id --port <port>`. You will have to replace "<port>" with the actual port connected to the NSPanel. This will do a check and see if the tool can communicate with the NSPanel.
- Run `esptool.py --baud 921600 --port /dev/ttyUSB0 write_flash 0x0 merged_flash.bin`. You will have to replace "/dev/ttyUSB0" with the actual port connected to the NSPanel.

Info: On Windows it might be just "esptool" without the ".py" at the end.

Info: On Windows "/dev/ttyUSB0" will have to be replaced by something like "COM4". If using MacOS or Linux the port will be something similar to "/dev/ttyUSB0".

After the flashing is complete you can continue with NSPanel configuration.

2.4 NSPanel configuration

To configure the NSPanel, do the following:

- Power up the panel.

Warning: If you have flashed multiple NSPanels, power them up one at a time as they will all have the same WiFi access point name.

Info: The GUI file has not been flashed yet so there will not be any visible change on the NSPanel screen.

- Connect to the new WiFi network "NSPMPanel" when the panel has started. WiFi password is **password**.
- When connected to the new WiFi network, make sure your device does not disconnect because it doesn't detect internet access. Then open a browser and navigate to "http://192.168.1.1".

Info: There have been issues when using Android and the Chrome browser that it sometimes just shows a blank page. If this is the case, either use a different browser (E.g. Firefox) or another device with WiFi to access the web page.

- Enter a friendly name for the NSPanel.

Note: This name is only used to register the NSPanel the first time. After the panel has been registered the name has to be changed from the manager web interface.

- Enter WiFi name and password.
- Enter MQTT address and port. If you are using authentication for MQTT, enter username and password as well.

Note: If you are running the MQTT server as an add-on to Home Assistant, enter the IP-address of your Home Assistant server.

- Press the "Save" button on the bottom of the page. The panel will reboot and try to connect to the WiFi network.

Info: If the panel fails to connect to the WiFi network for three minutes it will revert back and start the access point again. It will periodically scan for the configured WiFi and, if it detects that the configured WiFi has come back, it will reboot and connect to it.

- Connect to your WiFi again and go to the NSPanel Manager web interface. If all things are working and setup correctly the panel should show up in the list of panels on the first page. You will then have to accept or deny the panel access into the manager.

Info: If this is a US NSPanel version then it has to be set in the panel settings. Press the name of the NSPanel in the list and check the "Is US panel"-checkbox.

- Flash the new GUI file to the panel by pressing the "Actions"-button on the right and select "Update screen".

Note: The flashing of the GUI file may be finicky and might require multiple tries before it succeeds. If it fails and reboots or you see a "System data error", just try again.

- Continue on to create new rooms and add entities to your configuration. Continue with the section on the web interface and how it work here.

3 NSPanel Manager web interface

This chapter describes all pages available in the web interface and all settings that can be made. The web interface also links to specific sections in this chapter for help information.

3.1 First page

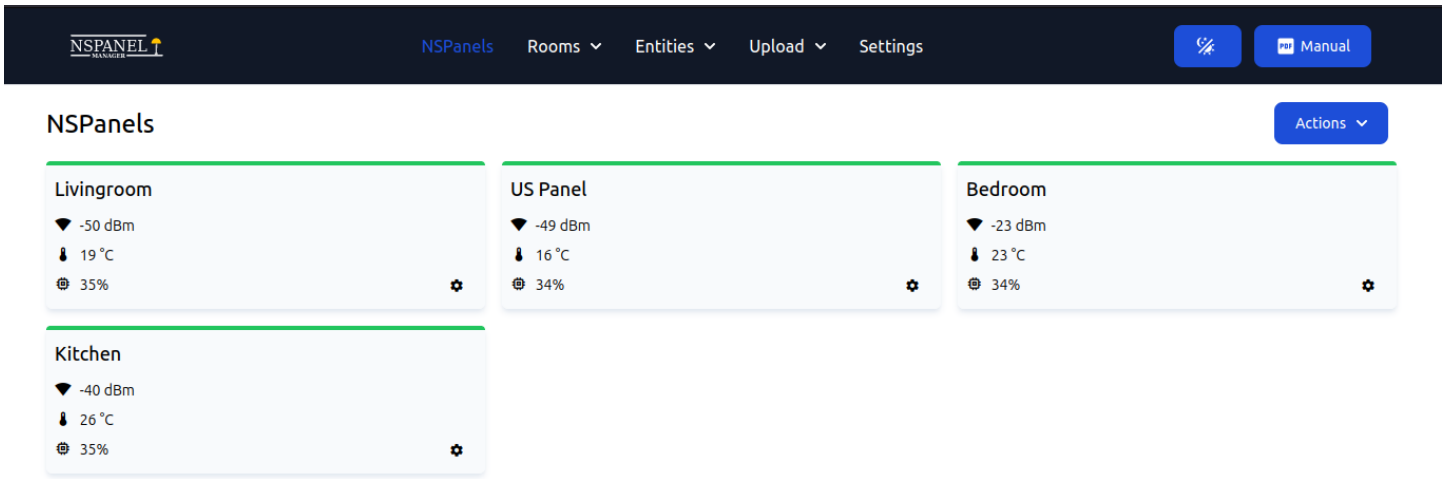


Figure 2: First page

The first page is used to get an overview of each registered NSPanel as well as perform actions on each panel individually or all at once. These actions are available in the menu accessed by pressing the cog. The available actions are:

- **Reboot** - Reboot the NSPanel.
- **Settings** - Navigate to the NSPanel settings page where individual settings for the panel can be made. This page can also be accessed by pressing the name of the panel.
- **Visit** - Visit the web page that is on the panel itself. This is the same page that was used to enter initial WiFi and MQTT information.
- **Firmware update** - Send a command to the NSPanel to update the firmware. The firmware will be downloaded from the manager.
- **GUI update** - Send a command to the NSPanel to update the screen. The screen firmware will be downloaded from the manager.
- **Delete** - Delete the nspanel from the manager.

There is also the "Actions"-button on the top right where you can perform some of the actions from the list above to all panels which are online. There is also an option to **Restart MQTTManager** which will restart the core component that handles all communication between different components of the NSPanel Manager software.

3.2 NSPanel page

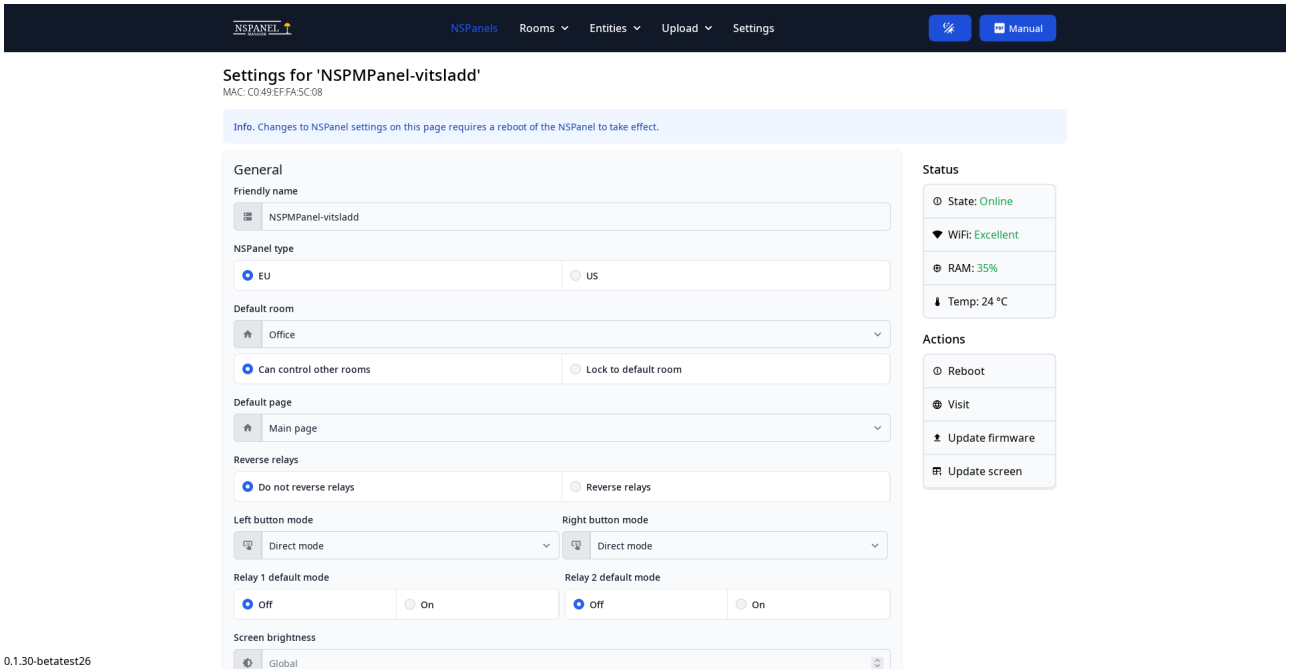


Figure 3: NSPanel page

The NSPanel settings page is used to configure specific settings for each individual NSPanel. This includes things like NSPanel type (EU/US), the room it is installed in, name and so on. See below for a complete list of options. The page can also be used to look at the live stream of logs from the NSPanel and send commands to the panel. What is displayed in the log view is dependent on what log level is configured on the settings page on the panel itself.

3.2.1 Available options

Available options for each individual NSPanel are:

- **Friendly name** - The name of the NSPanel.
- **NSPanel type** - The type of NSPanel, either **EU** or **US**.
- **Default room** - Select the default room for the NSPanel. This is the room that will be selected in the GUI after the NSPanel has booted and after the screensaver has been active. It is also possible to lock the NSPanel to the default room so that it can't control entities in other rooms.
- **Default page** - Select a different default page. Available options are:
  - **Main page** - The default page with sliders and buttons for light control.
  - **Scenes page** - The page where scenes can be activated and saved.
  - **Room page** - The page that is used to control individual entities.
- **Reverse relays** - Reverse the relays. This is the same as switching places of the cables on the back of the NSPanel. All actions meant for relay 1 will be applied to relay 2 and vice versa.
- **Left/Right Button mode** - Change the mode of a button. Available options are:
  - **Direct mode** - The buttons controls the relay directly. No WiFi is needed.
  - **Detached mode** - Detached mode controls a button via you selected home automation platform. This also exposes the option "Left/Right button controls light" where you select which entity it controls.
  - **Custom MQTT** - This mode can be used to send a custom message on a custom MQTT topic.

  - **Follow mode** - When the button is pressed the relay is engaged, when the button is release the relay is disconnected.

- **Relay 1/2 default mode** - Select the default mode of the relay when the NSPanel starts.
- **Screen brightness** - The brightness of the screen when on (1-100%).
- **Screensaver brightness** - The brightness of the screen when the screensaver is active (0-100%). A value of 0 means that the screen is off.
- **Screensaver mode** - Select what to display on the screensaver. Available options are:
  - **Global (use global setting)** - Use whatever is set in the "settings" section of the NSPanel Manager.
  - **Show with background (clock, date and weather)** - Show the screensaver with background image with current and forecast weather, time and date.
  - **Show without background (clock, date and weather)** - Show the screensaver with a black background with current and forecast weather, time and date.
  - **Show with background (no weather)** - Show the screensaver with background image but only time and date.
  - **Show without background (no weather)** - Show the screensaver with a black background but only time and date.
  - **No screensaver (turn off screen)** - Make the screen completely black when not in use.
- **Temperature calibration** - Calibrate the internal temperature sensor. Enter a negative value to bring the reported temperature down and vice versa.

3.3 Room page

This section will describe how to manage rooms. Most of the configuration done with NSPanel Manager will be done in rooms, please read this chapter for a full understanding on how to work with rooms.

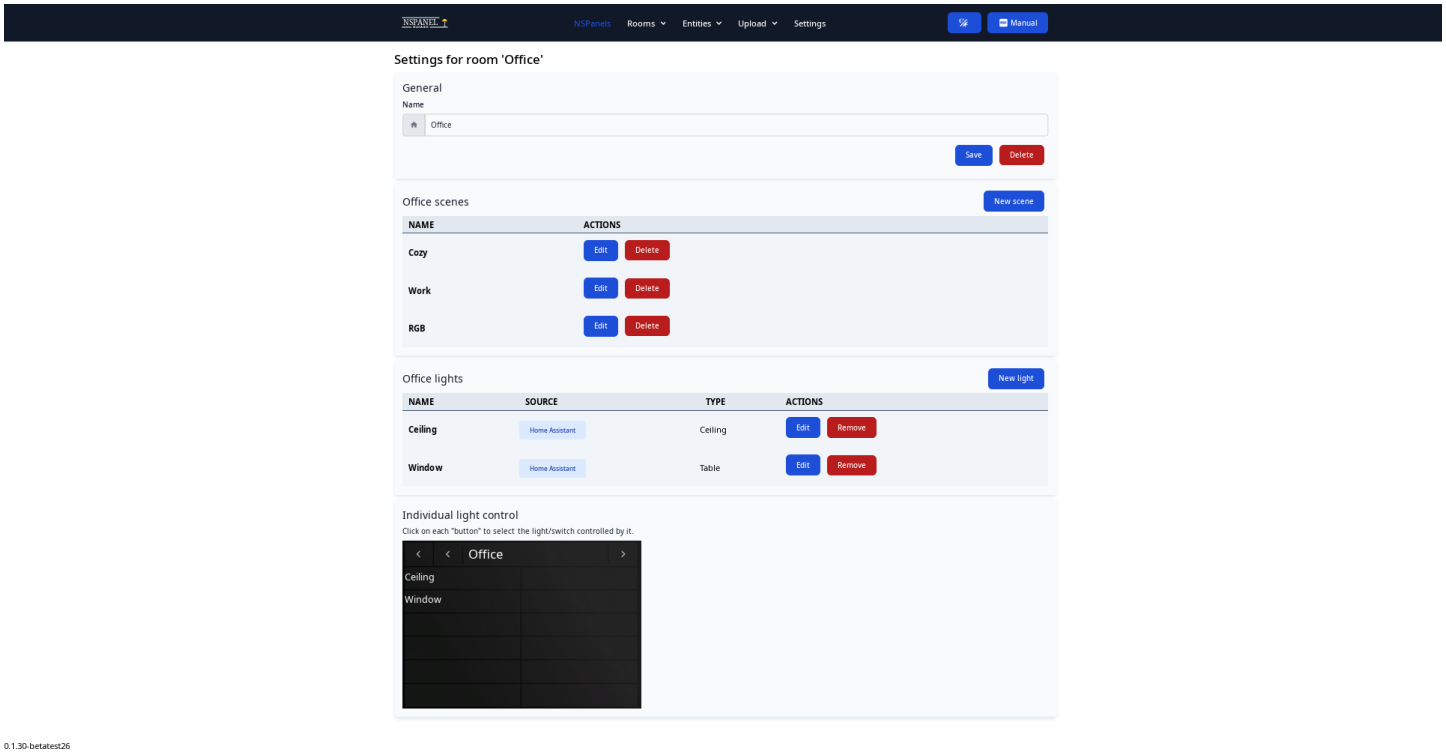


Figure 4: Room page

3.3.1 Scenes

At the moment, only NSPanel Manager scenes are available. They are easy enough to use. Simply create a scene in the room page and they will be available in the "Scenes"-list for the room on all NSPanels.
To save a scene, on the NSPanel hold the save button for the scenes for 3 seconds to save all the states of lights **currently added to the room**.
To recall/activate a scene, on the NSPanel press the name of the scene and all the saved light states will be restored for that scene.

Note: If a light was added after a scene was saved, that light is not affected by that scenes until the scene is saved again.

3.3.2 Lights

To add a new light, simply press the "Add new light"-button. When doing so, a list of all lights and switches gathered from Home Assistant and OpenHAB will be shown. Simply search or scroll to find the desired light and press it.

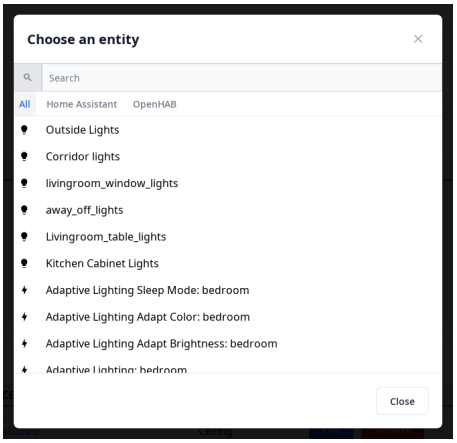
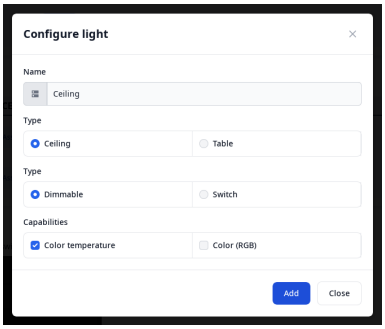
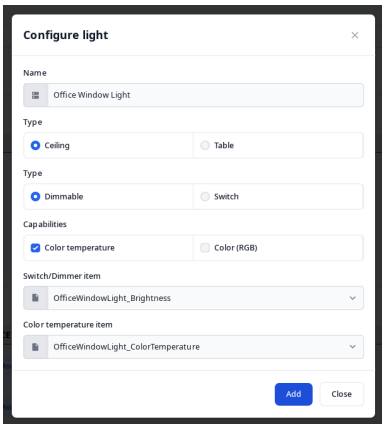


Figure 5: Adding a new light to a room

When done, a new screen will show up and depending on if the selected entity was from Home Assistant or OpenHAB difference options will be shown.



(a) Home Assistant



(b) OpenHAB

Figure 6: Add/Edit light

When adding a Home Assistant entity, simply set a friendly name for it, select the type (Ceiling or Table light), select if it's a switch or dimmable light and what other capabilities it has.
If you are adding an OpenHAB light or switch, things aren't as simple unfortunately. There is really no way around this and

as such the user has to chose all the same settings as for Home Assistant but also has to select the appropriate OpenHAB items that corresponds to each capability of the light.

3.3.3 Individual light control

There is place for up to 12 lights (per room) to be controlled individually from the NSPanel. The image on the bottom shows a preview on how this might look. When a new light is added to the room it will automatically be assigned to the next free slot on the page. By pressing a slot with an assigned entity you can chose to assign a new entity (if any entity is unasigned) or "clear" the slot which will remove the light from the page but it will still be attached to the room.

Info: Each entity may only be assigned to one slot. If the list of entities is empty then all entities has been assigned a slot.

3.4 Relay groups

Relay groups are used to bind multiple NSPanels and relays together so that when one relay in the group changes state, all the relays in the group transission to the new state.

3.5 Weather and Time

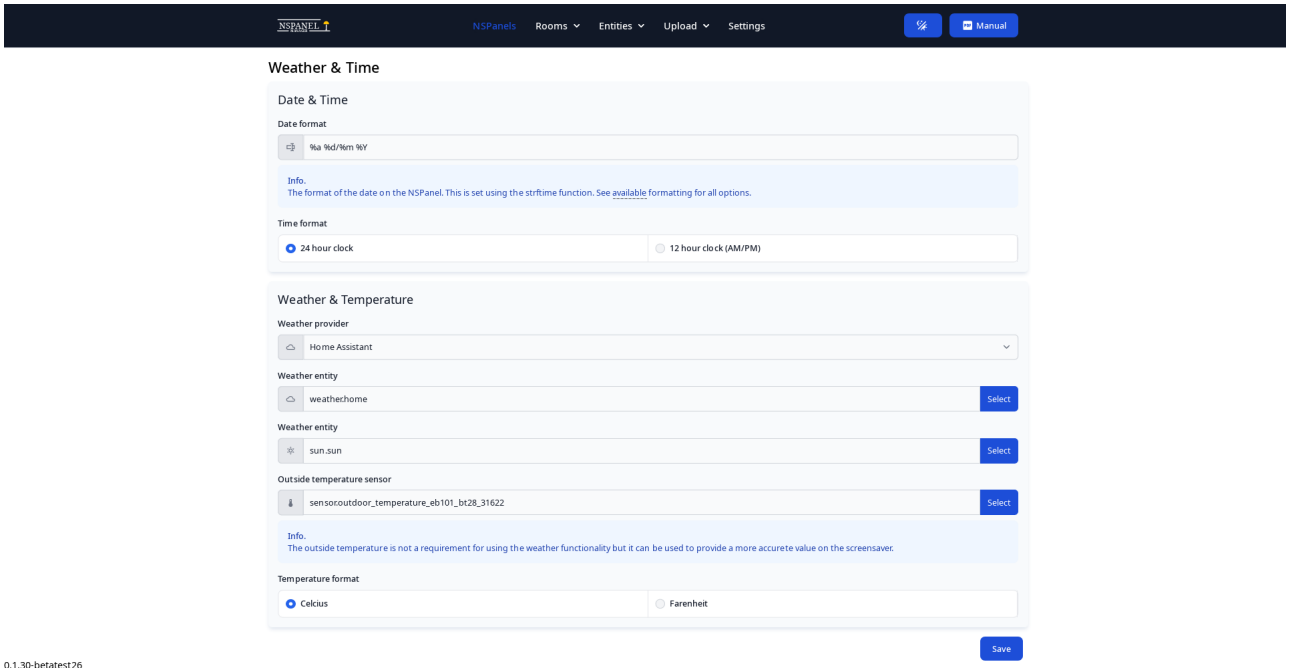


Figure 7: Weather and Time settings

On the Weather & Time page settings related to time, date and weather can be changed. Date format is configured according to the strftime function and a link to available formatting options is availabel in the information box.
It is also possible to select between 24 and 12-hour clock format.
To get the weather to work it needs to be configured in Home Assistant or OpenHAB. Please follow relevant text below:

3.5.1 Home Assistant

In order to use weather from Home Assistant, the following requirements has to be met:

- In Home Assistant, at least 1 weather provider is configured and working.
- In Home Assistant, at least 1 sun entity is available.

When the above requirements is met, press the "Select weather entitiy"-button and select an entity of type weather from Home Assistant. Home Assistant doesn't provide sun state in the weather entity and this has to be setup manually, press the "Select sun entity"-button and select an entity of type "sun". Press the save button to apply the settings.
If you wish to show temperature from a local temperature sensor available in Home Assistant, this can be selected in the **Outside temperature sensor** but it is not required.

3.5.2 OpenHAB

If using OpenHAB, there are a few things that has to be setup in OpenHAB before configuration in NSPanel Manager can happen. First of all, a weather entity for current weather and weather forecast has to be setup. NSPanel Manager only has support for AccuWeather and so, this guide will show how to setup AccuWeather in OpenHAB. First of all, create an account at AccuWeather. Once that is complete, you need to create an API-key, this can be done at this page. Give it a good name, something like OpenHAB so you in the future know what it's for. You do not need the "Core Weather" or "MinuteCast" products. Select "Weather app" for type of application, "C++" as language, select "Buissness to consumer", and lastly select your country.
Next we need to figure out what location ID to use to gather weather information for your location. The easiest way of doing this is to go to AccuWeather and locate your city. Once that is done, locate the weather ID in the URL. In the follow example URL the weather ID is 314929:

```
https://www.accuweather.com/en/se/stockholm/314929/weather-forecast/314929?city=stockholm&
    ↪ postalcode=undefined
```

Next, in OpenHAB you need to install the HTTP binding from the Add-on Store. After that is done, create a new .thing-file in the "things"-directory for your OpenHAB configuration and paste the following:

```
Thing http:url:accu_weather "Accu Weather" [
  baseURL="http://dataservice.accuweather.com/",
  refresh=1800] {
  Channels:
  Type string : weather "weather" [ stateExtension="/currentconditions/v1/<locationid>?apikey=<
      ↪ apikey>&details=true" ]
}

Thing http:url:accu_forecast "Accu Forecast" [
  baseURL="http://dataservice.accuweather.com/",
  refresh=43200] {
  Channels:
  Type string : forecast "forecast" [ stateExtension="/forecasts/v1/daily/5day/<locationid>?apikey
      ↪ =<apikey>&details=true&metric=true" ]
}
```

Info: Accu Weather only supports 50 API calls/day.There for the first thing (current weather) is set up to one call every 1800 seconds which gives 1 call every 30 minutes, 48 calls per day in total. The second thing (weather forecast is set up to on call every 43200 seconds which is 2 calls/day. In total this will maximize the API calls allowed to Accu Weather API.

Info: In case the HTTP binding does not initialize, you may need to increase the org.openhab.webclient:maxThreadsShared-value and the org.openhab.webclient:maxThreadsCustom-value in the conf/services/runtime.cfg-file for OpenHAB.

Info: If you do not want metric units, replace metric=true with metric=false in the stateExtension.

When you have pasted the above into a .things-file, you need to replace <locationid> with your location ID that we found above and replace <apikey> with your API-key from AccuWeather.
Once this is done and the weather HTTP-binding is showing up in OpenHAB you can press the "Select weather entity" in the NSPanel Manager web interface, select your weather entity and then select the channels for current weather and forecast.
If you wish to show temperature from a local temperature sensor available in OpenHAB, this can be selected in the **Outside temperature sensor** but it is not required.

3.6 Global settings

These settings will apply to all NSPanels (if they do not have specific configurations), and the NSPanel Manager container. There are two things that need to be set in order to get up and running. Those are:

1. Connection details to the same MQTT server that were set in the NSPanel configuration.
2. Connection details to Home Assistant and/or OpenHAB.

Important: Failing to meet both requirements listed above will result in a non-working setup!

Info: If running the NSPanel Manager container as a Home Assistant add-on then the Home Assistant connection details will already be configured.

There are also other settings that might be worth taking a look at while here, such as global scenes that apply to all entities, showing a clock on the screensaver, how bright the NSPanels should be, the Min & Max of color temperature and so on. Go ahead and explore by yourself.

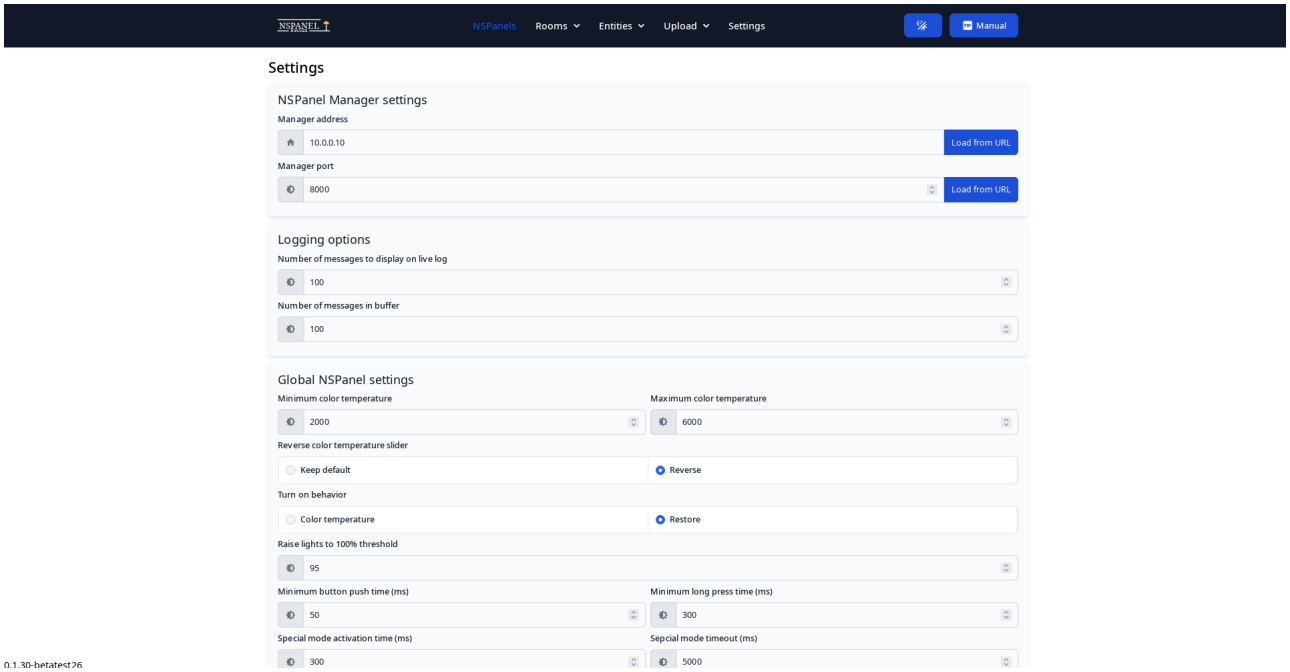


Figure 8: Global settings

The "Global settings" section of the web interface is where you can settings that apply to all NSPanels or settings that simply are not directly related the the panels themsevles. The following options are available: **Connection settings**

- **Manager address** - The is the IP address that will be sent to each NSPanel when it boots. This address is used in the NSPanel to communicate with the manager and has to be reachable from each and every NSPanel.
- **Manager port** - Port to use when connecting to the above mentioned IP address.

Logging options

- **Number of messages to display on live log** - The number of log messages to display in the live log for each NSPanel in the NSPanel settings page.
- **Number of messages in buffer** - The number of log messages to keep in buffer in the MQTTManager.

Global NSPanel settings

- **Minimum color temperature** - The minimum kelvin to send to Home Assistant/OpenHAB.

- **Maximum color temperature** - The maximum kelvin to send to Home Assistant/OpenHAB.
- **Reverse color temperature slider** - Reverse the color temperature sliders on the NSPanel used to control color temperature on lights.
- **Turn on behavior** - Wether to send color temperature when turning on the light or to simply send the "turn on" command.
- **Raise lights to 100% threshold** - The slider on the NSPanel can be finiky to get to reach the maximum value of 100%. To solve this problem this option can be used to force the value to be 100% above the given value.
- **Minimum button push time (ms)** - The minimum amout of time (in milliseconds) for the physical buttons to be pressed for it to register as a button press.
- **Minimum long press time (ms)** - The minimum amout of time (in milliseconds) for the physical buttons to be pressed for it to register as a long button press.
- **Special mode activation time (ms)** - The minimum amout of time (in milliseconds) to press the "table" or "ceiling" light button on the main page before it locks to control only table or ceiling lights.
- **Special mode timeout (ms)** - How long to wait (in milliseconds) after the special mode has been used before releasing and returning the operate on all lights.
- **Screen brightness (%)** - The default screen brightness when the NSPanel is used, 1-100%.
- **Screensaver brightness (%)** - The default screensaver brightness when the NSPanel is not used, 0-100%.
- **Screensaver activation timeout (ms)** - Amount of time (in milliseconds) to wait before turning on the screensaver.
- **Screensaver mode** - Default screensaver mode. Available options are:
  - **Show with background (clock, date and weather)** - Show the screensaver with background image with current and forecast weather, time and date.
  - **Show without background (clock, date and weather)** - Show the screensaver with a black background with current and forecast weather, time and date.
  - **Show with background (no weather)** - Show the screensaver with background image but only time and date.
  - **Show without background (no weather)** - Show the screensaver with a black background but only time and date.
  - **No screensaver (turn off screen)** - Make the screen completely black when not in use.
- **MQTT ignore time (ms)** - Time to wait (in milliseconds) after sending a command over MQTT before allowing new values to update the screen. This can be raised or lowered to minimize flickering of sliders on the NSPanel.

MQTT

- **MQTT address** - The address used to connect to the MQTT server.
- **MQTT port** - The port used to connect to the MQTT server.
- **MQTT username** - Username used to authenticate to the MQTT server. Leave empty if not used.
- **MQTT password** - Password used to authenticate to the MQTT server. Leave empty if not used.

Home Assistant API

- **Home Assistant address** - The address used to connect to Home Assistant. This should include "http://" or "https://" in the beginning and port are the end.
- **Access token** - The long lived access token used to authenticate to Home Assistant.

OpenHAB API

- **OpenHAB address** - The address used to connect to OpenHAB. This should include "http://" or "https://" in the beginning and port are the end.
- **Access token** - The access token used to authenticate to OpenHAB.

4 Upload

You may have noticed that there is an "Upload" menu in the top menu bar. Below which there exists items for uploading new firmware, data file and also .tft files. This menu is not something you will need to use unless instructed to. The items are used to upload new firmware, data files or .tft-files for testing. It is mainly used for development and debuging.

5 Panel functions

5.1 Main page

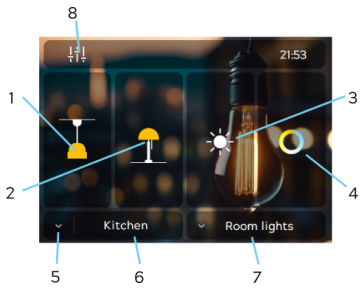


Figure 9: Main page

1. Ceiling Lights button
Short press - Toggle Ceiling lights ON/OFF.
Long press - Enter lock mode, sliders will now only effect ceiling lights. Short press to exit lock mode.

2. Table Lights button
Short press - Toggle Table lights ON/OFF
Long press - Enter lock mode, sliders will now only effect table lights. Short press to exit lock mode.

3. Brightness Dimmer
Control over Ceiling and Table Lights group. (More under Lights control logic below)

4. Kelvin Dimmer
Control over Ceiling and Table Lights group. (More under Lights control logic below)

5. Room toggle button
Short press - Change room.

6. Room button
Short press - Enter room page for individual device control.

7. Lights mode button
Short press - Toggle between Room Lights mode and All Lights mode

8. Scenes button.
Short press - Enter Scenes page. If 7 is in Room Lights mode user will enter Room Scenes. If 7 is in All Lights mode user will enter Global Scenes page.

5.1.1 Lights control logic

The NSPanel main page might have some behavior that seems odd at first but the logic of it will be described here. The first page will affect entities in the selected room (if in room-mode) or all the configured lights (if in "All lights"-mode). The two left buttons for ceiling and table lights will always behave the same. Pressing a button that is "off" will turn on all the lights of that type. Pressing a button that is "on" will turn off all lights of that type. The sliders will always display an average value of all entities that will be affected of changes. There are few different scenarios:

One or more lights on
When changing the sliders, the changes will only be sent out to the lights currently on. If turning on a group of lights or individual lights they will be turned on to the current brightness of the slider. I.e. average dimming level in the room.

Info: If you wish for the light to always turn on with color temperature even though you turned it off from RGB, there is a setting in the global settings.

No lights on
When changing the sliders, the changes will be sent out to all lights selected (depending on room or "all lights"-mode).

Lock mode
You can lock which light to affect by pressing and holding either the ceiling or table-lights button. This will enter a special mode where changes to the sliders will only affect the selected type of lights. By pressing the same button again you can exit the "special mode". The "special mode" will also time out after a few seconds.

5.1.2 Scenes button (top left corner)

The settings icon in the top left corner is for entering the Scenes page. In NSPanel Manager there are both Room Scenes and Global Scenes. If you're in Room Lights mode (button in lower right corner) you will enter the Room Scenes page. If you're in All Lights mode you will enter the Global Scenes page. You'll also see that the settings icon changes when toggling between Room Lights and All Lights mode. Standard settings icon leads to Room Scenes page when pressing it. Settings icon with a roof on top leads to Global Scenes page.

5.1.3 Swipe down

You can swipe downwards when on the first page to enter the Smart Home Control page. This page is a work in progress. Design and funcionality is not finished or decided yet.

5.2 Smart Home Control page

Accessed by swiping down on Main page. Work in progress. Design and funcionality is not finished or decided yet.

5.3 Scenes page

To enter Scenes page, press the settings icon in top left corner on Main page. Depending on if you're in Room Lights mode or All Lights mode you will enter Room Scenes page or Global Scenes page.
Scene names that show up here are the ones you have configured in the NSPanel Manager web interface. To store a scene simply hold the save button for three seconds and the current values of the lights in the room you are in or all lights if in All Lights mode will be saved. To activate a scene and send out those saved values you just press the scene name.

5.4 Room page

Enter Room page by pressing the room name on Main page. All devices configured for that room will show up here. To control a device individually press the device name.

5.5 Individual Lights page

All the capabilities of the chosen light will be shown on this page. If the light is RGB capable there will be an icon in the top right corner to toggle between Color Temperature mode and Color mode.´

6 Logs

While logs are normally sent over MQTT, any logs that are created before WiFi-connection are sent out on Serial. If you wish to see the logs going over MQTT, you can look at the topic `nspanel/<panel name>/log`. If you wish to look at the logs going over serial, you can use programs like Putty. Connect to the NSPanel with the serial programmer as usual but **dont't** connect IO0 to GND. In Putty enter your serial port in the "Serial line" box and choose baud 115200. You should then be able to connect by pressing the "Open"-button. Example:

Info: On Windows "/dev/ttyUSB0" will have to be replaced by something like "COM4". If using MacOS or Linux the port will be something similar to "/dev/ttyUSB0".

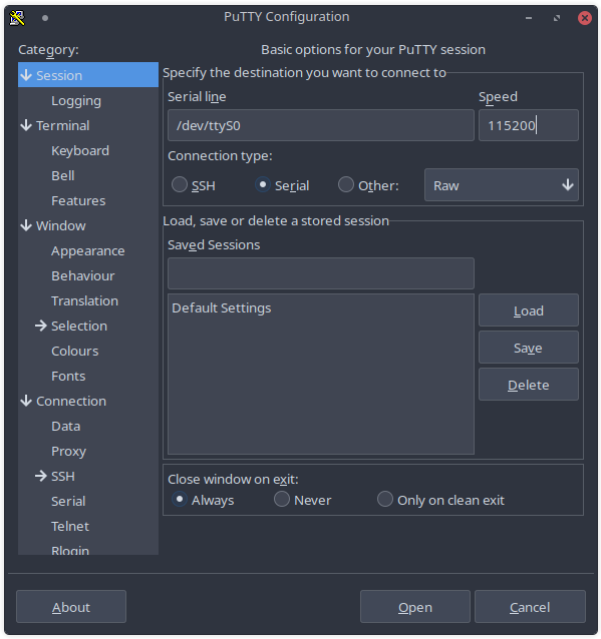


Figure 10: Connecting to Serial with Putty

7 Advanced setup

7.1 Manual Docker container setup

If you wish to manually build and setup the Docker container from source first clone/download this git repository. Then use the command `docker build -t nspanelmanager .` while standing in the downloaded "docker"-directory. This will always be the same. To then start the container, the following command can be used:

```
docker run --name nspanelmanager -e 'TZ=Europe/Stockholm' -v \
"$(pwd)/data/":"/data/" \
-d -p 8000:8000 nspanelmanager
```

To change the timezone, set the TZ-environment variable, for example: `-e TZ=Europe/Stockholm`.

Important: All data for NSPanel Manager is stored in the directory mapped to "/data" in the container. In this case, the "data"-directory where you are currently standing.

7.2 Docker-compose container setup

If you wish to run NSPanel manager from docker compose you can use the below example as a template for your setup.

Note: This example is for an x86_64 machine placed in the Europe/Stockholm timezone. Replace image name and timezone as needed.

```
services:
  nspanelmanager:
    image: nspanelmanager/nspanelmanager-amd64
    container_name: nspanelmanager
    environment:
      - TZ=Europe/Stockholm
    volumes:
      - /nspmdata/:/data/
    ports:
      - 8000:8000
    restart: always
```

Important: All data for NSPanel Manager is stored in the directory mapped to "/data" in the container. In this case, the "/nspmdata/"-directory where you are currently standing.

8 Functional information

8.1 Software components

There are really three software components written for the NSPanel Manager project. These are described as below:

- **Web interface:** The web interface that you interact with is built on top of the Django framework. This software gives the user an interface to interact and configure the project with. This software also manages the database with settings.
- **MQTT Manager:** There is a second software running in the background on the NSPanel Manager container that hosts the web interface. This component is named "MQTTManager". The MQTTManager handles all things with MQTT. It loads the config from the web interface via the API and then processes all commands from panels, state updates from Home Assistant and OpenHAB and so on. It's basically the glue that makes the panel's actions affect Home Assistant and OpenHAB. The MQTTManager is also the software that send state updates from Home Assistant and OpenHAB to the panels when changes occur.
- **The NSPanel firmware:** The firmware written for the NSPanel has been specifically designed to be as response and easy to use as possible. The firmware handles all communication with the TFT (Nextion) display and with MQTTManager via MQTT.

8.2 Data flow

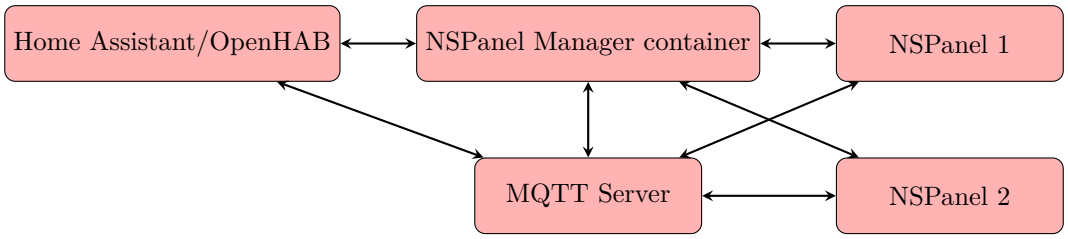


Figure 11: NSPanel Manager data flow

The data flow within NSPanel Manager might look intimidating but it's not that bad. Below is an explanation of all the arrows above.

8.2.1 Home Assistant and/or OpenHAB to/from NSPanel Manager container

There is two types of traffic flowing between these nodes:

- **Websocket:** A websocket connection is setup in order for the NSPanel Manager container to receive entity updates from Home Assistant and/or OpenHAB but also to sent entity commands (E.. turn light X on to 20%). A websocket is used to speed up the communication and also to not have to poll the home automation software for information.
- **HTTP GET API:** The usual HTTP GET API is also used. This is used when adding entities to a room, as an example. When pressing the "Add new light" button, the NSPanel Manager container will make an HTTP GET request to gather all available entities and then send them back to the client (browser) so that the user may choose what entitiy to add to the room.

8.2.2 NSPanel Manager container to/from MQTT

MQTT is used to send updated entity states received from the home automation software out to all NSPanels and also receive states and commands from NSPanels.

8.2.3 Home Assistant and/or OpenHAB to/from MQTT

Home Assistant and OpenHAB can leverage the MQTT integration through "Home Assistant MQTT Auto-discovery" (which OpenHAB can also use) to auto-discover NSPanels and automatically register entities for panel temperature reading, panel relays, screen state and so on.

8.2.4 NSPanel Manager container to/from NSPanels

The configuration of lights, scenes and so on does not reside on each panel. The panel only has localy the bare minimum configuration for setup. When the panel starts and has connected to WiFi it will do a HTTP GET request to the NSPanel Manager container in order to receive all configuration of entities, screen brightness and really, all settings available in the NSPanel Manager web interface.

8.2.5 MQTT to/from NSPanels

Each NSPanel send states (E.g. temperature) and commands (E.g. turning on a light) over MQTT for the NSPanel Manager container to pickup. The panel also received commands, E.g. turn on relay 1, turn on screen and so on.

8.3 MQTT Topics

Below table is a description of all MQTT topics that might be of use by a user. Replace <panel_name> with the friendly name of your NSPanel:

Topic	**Payload**	**Description**
nspanel/<panel_name>/screen_cmd	1 or 0	Send a 1 or 0 to turn on/off the display.
nspanel/<panel_name>/screen_state	1 or 0	Current state of the screen.
nspanel/<panel_name>/brightness	1 to 100	Control the brightness of the screen.
nspanel/<panel_name>/brightness_screensaver	0 to 100	Control the brightness of the screensaver.
nspanel/<panel_name>/r1_cmd	1 or 0	Send a 1 or 0 to turn on/off relay 1.
nspanel/<panel_name>/r1_state	1 or 0	The current state of relay 1.
nspanel/<panel_name>/r2_cmd	1 or 0	Send a 1 or 0 to turn on/off relay 2.
nspanel/<panel_name>/r2_state	1 or 0	The current state of relay 2.
nspanel/<panel_name>/temperature_state	Current temperature	The current temperature reading.
nspanel/<panel_name>/log	Log message	The panel will send live logs on this topic.

There are more topics that are used internally, these are:

Topic	**Payload**	**Description**
nspanel/entities/<type>/<id>/state_<attribute>	The value of the attribute	An update of entity state value sent out by MQTTManager. Example: nspanel/entities/light/42/state_kelvin
nspanel/status/time	Time as a string	Current time sent by MQTTManager.
nspanel/status/weather	JSON	A JSON representation of the current weather and weather forecast.
nspanel/<panel_name>/status_report	JSON	JSON payload with current state of the panel.
nspanel/mqttmanager/command	JSON	JSON payload from panel with a command for MQTTManager to perform.