LOXONE

Structure File

Loxone Config 10.0

No Gimmicks.
Real Smart Homes.

The aim of this document is to give you a fundamental understanding on how the static structure of a Miniservers Configuration is represented. At Loxone we call this representation “Structure-File” and it is available as “LoxAPP3.json”.

In order to create a UI for remote controlling a Smart-Home different infos are required. There is the static structure which changes due to modifications in the configuration itself. On the other side there are the states that change over time due to the permanently changing environment (temperature, movement, ..) or actions taken.

While “Communicating with the Miniserver” did go into detail on how the dynamic states are structured and communicated, this document is going to focus on the fixed structure.

Table of contents

UpDownLeftRight digital

UpDownLeftRight analog

ValueSelector

Ventilation

Webpage

WindowMonitor

General Info

An Smart Home is a set of various sensors and actuators that are linked together by our Miniserver. Rooms and Categories are used to group these sensors and actuators (which we'll be calling controls from now on). And besides those controls and the rooms and categories which they belong to, there are some other global and external informations, like the weather-server or info about the Miniserver itself. These different types of informations lay out the basic structure of this file and in the next chapters, we're going to go into detail on each one of these.

Structure changes in 10.0

Intelligent Room Controller v2

An all new Room controller

Intelligent Room Controller Daytimer v2

This new Daytimer is a subcontrol of the new Intelligent Room Controller v2

ClimateController

The all new Climate Controller manage your heating or cooling sources.

SolarPumpController

Visualize all your warm water buffers with the new SolarPumpController

AlarmClock

AlarmClock has been extended to work with the new Touch Wake Light Air.

- **Details**
  - hasNightLight
  - brightInactiveConnected
  - brightActiveConnected
- **States**

    - entryList
        - Every entry has the new “nightLight” property which indicates, if it is a Touch Wake Light Air
    - deviceState
    - deviceSettings
- **Commands**
    - setBeepOn/{0 / 1}
    - setBrightnesInactive|{0 - 100}
    - setBrightnessActive/{0-100}

lastModified

Since the Structure-File can grow rather large, it would not be a good idea to download a completely new version each time the UI is built up. So once you download the Structure-File, make sure that you cache it and to save the “lastModified” attribute value. Every time you reconnect you can use the command “jdev/sps/LoxAPPversion3” to compare whether or not the Structure you’ve cached is still up to date.

msInfo

The msInfo area contains static information on the Miniserver and it’s configuration. While some of these are pretty self explanatory, the need for others might be unclear at first.

- serialNr
    - serial number of this Miniserver
- msName
    - Name of the Miniserver as specified in the configuration document
- projectName
    - Name of the configuration document used for this Miniserver.
- localUrl
    - IP & Port that are used to connect to this Miniserver inside its local network.
- remoteUrl
    - Url/IP & Port using which the Miniserver is globally reachable.
- tempUnit
    - Gives info what temperature unit is used by this Miniserver.
        - 0 = °C
        - 1 = °F
- currency
    - a string containing the currency symbol to use (€, $, …)
- squareMeasure
    - the unit of area (for rooms)
- location

    - The address of this Miniserver, where is it located. This info is also used for calculating the sunrise & sunset as well as for the cloud-weather.
- heatPeriodStart
    - **DEPRECATED**
        - Replaced by the command jdev/sps/calendargetheatperiod
    - Month and day when the heating period starts each year, used by the intelligent room-controller.
- heatPeriodEnd
    - **DEPRECATED**
        - Replaced by the command jdev/sps/calendargetheatperiod
    - End of the heating period.
- coolPeriodStart
    - **DEPRECATED**
        - Replaced by the command jdev/sps/calendargetcoolperiod
    - Start of the cooling period.
- coolPeriodEnd
    - **DEPRECATED**
        - Replaced by the command jdev/sps/calendargetcoolperiod
    - End of the cooling period.
- catTitle
    - Top level name for all the categories, maybe there is a different kind of grouping.
- roomTitle
    - Some configurations handle the location grouping differently, they might not want to call it “Room” but “Zone” or alike.
- miniserverType
    - 0 or missing → regular Miniserver
    - 1 → Miniserver Go
- sortByRating
    - Indicates whether or not the controls are to be sorted based on their rating, which was specified in the config using the stars (0-5).
- currentUser
    - name
    - uuid
    - isAdmin
    - changePassword
        - if the user is allowed to change the password via WebService

globalStates

This section lists all states that affect not only a single control but the whole Miniserver. The UUIDs here can be used to lookup the corresponding state values that arrive with all the other controls state updates.

- sunrise
  - seconds since midnight, time when the sun will rise at the Miniservers location.
- sunset
  - minutes since midnight, time when the sun will go down.
- favColorSequences
  - An array of color sequences that have been marked as favorites in any light controller v2.
- favColors
  - An array of colors that have been marked as favorites in any light controller v2
- notifications
  - Push Notifications are also sent to open WebSocket connections (if the user is allowed & registered for these notifications)
  - ```
    {
      “uid”: String,                  // unique message id
      “ts”: Number,                   // unix timestamp
      “type”: 10,                     // type, 10 = normal message
      “title”: String,                // title
      “message”: String,              // message
      “data”: {                       // additional data
        “lvl”: Number,                // level, 1 = Info, 2 = Error, 3 = SystemError
        [“uuid”: String]              // optional uuid (from Control, eg. Alarm)
      }
    }
    ```
- miniserverTime
  - the current date, time & UTC-offset of the Miniserver. (e.g. “2017-07-03 13:01:36 +02:00:00”)
  - Updates only when the time is changed (e.g. shift to/from daylight saving time or manual time change).

operatingModes

Operating Modes are used by the Miniserver to respond to time-based events, like a weekday, weekend, vacation or alike. Mostly this is used in daytimers and the intelligent roomcontrollers.
    ```

rooms

In this section, there's a list of all the rooms that are used to group the controls in the configuration.

- uuid
  - Unique identifier for this room on this Miniserver
- name
- image
  - Icon for this Room
- defaultRating
  - Based on this number, the rooms are sorted in the UI (depending on the sortByRating-Attribute in msInfo

cats

Just like the rooms that group controls based on their location, categories group them logically.

- type
  - Categories can be given a type, that provide semantic info on what the controls in this categorie are for.
    - lights
    - indoortemperature
    - shading
    - media
- color
  - categories can be given a color on the UI

weatherServer

If a Cloud Weather is configured, this section is added to the Structure-File.

states

There are only two states listed here, actual and forecast. Along with the state-updates of the other controls, the weather-updates are also delivered. See the corresponding section in the "Communicating with the Miniserver" document for details on how to parse the Weather-State-Events.

- actual
  - the current weather data for the moment

- forecast
    - the weather data for the future (for the next 96 hours)

format

This is a list on what (C-Style) formats to use for the different states.

- relativeHumidity
- temperature
- windSpeed
- precipitation
- barometricPressure

weatherTypeTexts

Each forecast and the actual weather situation has a type that is visualized differently. This section gives the user friendly texts for each of this weather situations.

weatherFieldTypes

available since Miniserver 8. Returns the possible weather field types. This types are the same as in the Loxone Config (e. g. Temperature)

times

available since Miniserver 8. Returns a list of possible time fields. This types are the same as in the Loxone Config (e. g. Minutes until sunset).

caller

available since Miniserver 8. Returns a list of configured caller service in the Loxone Config.

autopilot

available since Miniserver 8. This section is used for the autopilot generator configuration. An autopilot is a rule which can be created on the app side and is executed on the Miniserver as soon as all events of the rule are matched. The API isn't publicly available.

mediaServer

This section is present as soon as one or more Loxone Music Servers (or Castatunes Servers) are in use on this Miniserver. They are generically referred to as "mediaServer" and each one is identified by a UUID.

- type
    - 1 = Loxone Music Server
    - 0 = Casatunes
- host

- IP and port used for communicating with the server

Loxone Music Server

The Loxone Music Server provides a powerful API to clients that are connected directly to it, here are some examples:

- adding/removing and browsing external services like Spotify or Google Music
- browsing the music stored directly on the Music Server
- creating playlists
- browsing web-radio-stations
- modifying per zone favorites

This API is not covered in this documentation and is not publicly available as of now.

messageCenter

available since Miniserver 10.0. This section is used for the Systemstatus, the ultimative trouble guide for your Smart Home.

States

- changed
  - Unixtimestamp when the the Systemstate entries have been modified the last time

Controls

Controls are a term that covers both actuators and sensors, simple in- or outputs and complex block-controls like an intelligent roomcontroller.

Mandatory fields

- name
- type
  - this attribute identifies what kind of control this is (Jalousie, Daytimer, …)
  - an empty string as type indicates a control that should not be visualized.
- uuidAction
  - unique identifier for this control
- defaultRating
  - just like rooms and categories, controls can also be rated
- isSecured
  - whether or not the visualisation password has to be entered for this control

Optional fields

The optional fields might differ between the various types of controls as well as between different controls of the same type. More detailed information can be found in the documentation for the different types of controls.

- room
  - uuid of the room this control belongs to
- cat
  - uuid of the category this control belongs to
- states
  - list of state uuids for the control
- securedDetails
  - indicates that there is sensitive information available. for details see Secured Details
- details
  - visualisation details, like the format
- statistic
  - if a control is scheduled for recording its values to a statistic, this attributes contains all info necessary.

Statistic

Each control that has statistics enabled, will provide the following infos in its statistic attribute:

- frequency
  - how often is the statistic written
  - 1 = every change
  - 2 = average per minute
  - 3 = average per 5 minutes
  - 4 = average per 10 minutes
  - 5 = average per 30 minutes
  - 6 = average per hour
- outputs
  - an array of outputs for whom statistic data is recorded
  - each output has the following attributes
    - id = Index, what datapoint-row is used for this output (0-6)
    - name
    - format
      - format specifier for analog values
    - uuid

        - visuType
            - 0 = line chart
            - 1 = digital
            - 2 = bar chart

Commands

- jdev/sps/getstatsdate
    - date of statistics.json file
- statistics.json
    - contains info of all stored statistics
- statistics.json/{controlUUID}
    - filtered statistic file
    - available since 6.1.10.16
- binstatisticdata/{controlUUID}/{date}
    - returns a binary “stream”, format below
    - {date} = “YYYYMM” or “YYYYMMDD”

BinaryFormat

Via websocket binstatisticdata can be downloaded. Here’s how it is structured.

- ts: Uint32 (4 Bytes)
    - seconds since 1.1.2009 in local Miniserver-time
- values Float64 (8 Bytes)
    - amount of values can be found in the LoxAPP3.json for each Control (property “statistic”, length of “outputs” array)
    - if multiple outputs are available (eg. meter) the order is the same as in outputs array of the control statistic details

Secured Details

If the flag “securedDetails” is set, this indicates that a control has sensitive information available, such as credentials for accessing an Intercoms video stream. This sensitive information needs to be requested using a separate, encrypted request.

Commands

- jdev/sps/io/{controlUUID}/securedDetails
    - Will return all securedDetails of the control

Control Types

In the following sections you will find details on the different types for controls in our LoxAPP3.json. It will not provide a detailed documentation on how and what these controls are being used for. If you lack this info, please see our online documentation.

It will provide info on what commands these controls support and what states they will provide.

Alarm

Covered Config Items

- Burglar alarm

Details

- alert
  - Not used
- presenceConnected
  - TRUE if a presence detector is connected to DisMv

States

- armed
  - If the AlarmControl is armed
- nextLevel
  - the ID of the next alarm level
    - 1 = Silent
    - 2 = Acustic
    - 3 = Optical
    - 4 = Internal
    - 5 = External
    - 6 = Remote
- nextLevelDelay
  - The delay of the next level in seconds, this can be specified with the parameters D1 - D6 in Loxone Config. This increments every second...
- nextLevelDelayTotal
  - The total delay of the next level in seconds, this can be specified with the parameters D1 - D6 in Loxone Config.
- level
  - The ID of the current alarm level
    - 1 = Silent
    - 2 = Acustic
    - 3 = Optical
    - 4 = Internal
    - 5 = External
    - 6 = Remote

- armedDelay
  - The delay of the alarm control being armed
- armedDelayTotal
  - The total delay of the alarm control being armed
- sensors
  - A string of sensors separated by a pipe (“|”)
- disableMove
  - If the movement is disabled or not
- startTime
  - timestamp when alarm started

Commands

- on
  - Arms the AlarmControl
- on/{number}
  - number can be 0 or 1
    - 0 = arm without movement
    - 1 = arm with movement
  - available since Miniserver 7.4.4.14
- delayedon
  - Arms the AlarmControl with the given delay (Parameter “Da”)
- delayedon/{number}
  - number can be 0 or 1
    - 0 = delayed arm without movement
    - 1 = delayed arm with movement
  - available since Miniserver 7.4.4.14
- off
  - Disarms the AlarmControl
- quit
  - Acknowledge the alarm (quit the alarm)
- dismv/{number}
  - number can be 0 or 1
    - 0 = disable movement
    - 1 = enable movement

AlarmClock

Operating mode 0, 1 and 2 are prioritized!

Covered Config Items

- Alarm clock

Details

- hasNightLight
  - If the control has a Touch Nightlight
  - Available since Miniserver 9.3
- brightInactiveConnected
  - Is the connector for the nightlight brightness inactive connected
  - Available since Miniserver 10.0
- brightActiveConnected
  - Is the connector for the nightlight brightness active connected
  - Available since Miniserver 10.0

States

- isEnabled
  - If the AlarmClock is enabled
- isAlarmActive
  - If an entry is ringing
- confirmationNeeded
  - If the User needs to confirm the entry
- entryList
  - Object with all the entries
    - {
      - “entryID”:
        - {
          - “name”: “AlarmClock1”,
          - “isActive”: true,
          - “alarmTime”: 29940,
          - “modes”: [0,1,2,3,5],
          - “nightLight: false,
          - “daily”: false
        - }
    - }
    - nightLight and daily are available since Miniserver 9.3
- currentEntry
  - The “entryID” of the current entry
  - 0 if there is no current entry
- nextEntry
  - The “entryID” of the next entry
  - 0 if there is no next entry
- nextEntryMode

    - Represents operating modes 3 - 9 from our structure file
- ringingTime
    - countdown in seconds how long the alarmClock will be ringing until it's going to snooze again
- ringDuration
    - The duration the AlarmClock is ringing
- prepareDuration
    - The preparation time in seconds
- snoozeTime
    - Seconds until snoozing ends
- snoozeDuration
    - Duration of snoozing
- nextEntryTime
    - Date of next entry in seconds since 1.1.2009
    - available since Miniserver 8.1
- deviceState
    - 0 = No Touch Nightlight is connected
    - 1 = Touch Nightlight is offline
    - 2 = Touch Nightlight is online
    - available since Miniserver 9.3
- deviceSettings
    - json object that is empty when no nightlight is used
    - available since Miniserver 10.0
    - {
    - "beepUsed":true, // BOOLEAN if the Buzzer on the nightlight is used
    - "brightInactive":0, // value - brightness of display when inactive
    - "brightActive:100 // value - brightness of display when active
    - }

Commands

- snooze
    - snoozes the current active entry
- dismiss
    - dismisses the current active entry
- setActive/{active}
    - 1 = activate 0 = deactivate
    - This activates or deactivates the AlarmClock
- entryList/put/{entryID}/{name}/{alarmTime}/{isActive}/{modes}
    - Creates an entry or overrides it if the entryID is the same
    - entryID
        - The ID of the entry (Existing IDs will be overridden)

        - name
            - The name of the entry
        - alarmTime
            - Alarmtime in seconds since midnight
        - isActive
            - If the entry should be activated per default
        - modes
            - an array of mode IDs (operating modes found in our structure file)
                - Example: “1,3,5,6,7” -> “Holidays, Mondays, Wednesday, Thursday, Friday”
- entryList/delete/{entryID}
    - Deletes an entry with the same entryID
    - entryID
        - The ID of the entry
- setPrepDuration/{number}
    - Sets the prepare duration
        - number = The Prepare duration in seconds
- setRingDuration/{number}
    - Sets the ringing duration
        - number = The Ringing duration in seconds
- setSnoozeDuration/{number}
    - Sets the snoozing duration
        - number = The snoozing duration in seconds
- setNeedsConf
    - If the user needs to dismiss or snooze the alarm
- setBeepOn/{1 / 0}
    - Set if the buzzer on the nightlight should be used
- setBrightnessInactive/{0-100}
    - Set the display brightness of the nightlight when inactive
- setBrightnessActive/{0-100}
    - Set the display brightness of the nightlight when active

AudioZone

Covered Config Items

- Music Server Zone

Details

- server

    - the UUID of the Loxone Music Server this zone belongs to. See section on Loxone Music Server for details.
- playerid
    - the ID used to identify this zone within the Loxone Music Server
- clientType
    - 0 = physically connected
    - 1 = UPNP

States

- serverState
    - -3 = unknown/invalid zone
    - -2 = not reachable
    - -1 = unknown
    - 0 = offline
    - 1 = initializing (booting, trying to reach it)
    - 2 = online
- playState
    - -1 = unknown
    - 0 = stopped
    - 1 = paused
    - 2 = playing
- clientState
    - only used for UPNP clients!
    - 0 = offline
    - 1 = initializing (booting, trying to reach it)
    - 2 = online
- power
    - whether or not the client power is active
- volume
    - current volume
- maxVolume
    - zones can be assigned a maximum volume
- volumeStep
    - how large a single volume step is (important for button-control)
- shuffle
    - shuffle/0 = off
    - shuffle/1 = on
- sourceList
    - JSON containing all zone-favorites.
    - e.g.: { "getroomfavs_result": [{"id": 3, "type":4, "totalitems":6, "start":0, "items": [{ "slot": 1, "name": "Led Zeppelin" }, { "slot": 8, "name": "Stüberl" }, { "slot": 7,

"name": "Dein Mix der Woche" }, { "slot": 2, "name": "07 Interlude 2.mp3" }, { "slot": 5, "name": "Johnny Cash" }, { "slot": 3, "name": "Große Liste" }] }], "command":"audio/cfg/getroomfavs/3/0/10" }

- repeat
  - -1 = unknown
  - 0 = off
  - 1 = repeat all
  - 2 = -not used-
  - 3 = repeat current item
- songName
- duration
  - how long the whole track is, -1 if not known (stream)
- progress
  - current position in the track, will be updated every 10 seconds
- album
- artist
- station
- genre
- cover
  - path to an image representing the current item.
- source
  - current selected source identifier (integer)
  - available since Miniserver 7.4.4.14, Music Server 1.1.4.14

Commands

- volume/{newVolume}
- volstep/{step}
  - increases or decreases the current volume by a step. E.g.” volstep/+3” or “volstep/-3”
  - available since Miniserver 8.0
- prev
  - previous track
- next
  - next track
- play
  - (urns the client on if needed
- pause
- progress/{seconds}
- shuffle
  - toggles the shuffle state on/off
- repeat/{repeatState}

    - 0 = off
    - 1 = repeat list
    - 3 = repeat track
- on
    - turns the client on and starts playing right away
- off
    - turns the client off
- svpower/on
    - will wake the Music Server from standby
- svpower/off
    - will send the Music Server into standby
- source/{sourceNumber}
    - 1-8, starts playing the corresponding zone-favorite as specified by the app.
    - as of now, the list of zone-favorites cannot be obtained from the Miniserver via this API

CarCharger

Covered Config Items

- Wallbox

Details

- chargerType
    - -1 = no external charger, block only
    - 0 = KEBA
    - 1 = BMW
- limitAllowed
    - whether or not the charging limit input is used. If false, limitMode 2 (Autopilot) will default to limitMode 0 (maxLImit)
    - available since Miniserver 8.0

States

- status (0=Offline, 1=Initializing, 2=Online)
- charging (0,1)
- connected (0,1)
- chargingFinished (0,1)
- power (kW)
- energySession (kWh)
- limitMode (0=Off, 1 = Manual, 2= Autopilot)
- currentLimit (kW)

- minLimit (kW)
- maxLimit (kW)
- chargeDuration (Secs)
- showLoadManagement (0,1)
    - Standalone, Keba
        - always 1
    - BMW
        - “Loadmanagement with Miniserver” must be enabled on the BMW Wallbox

Commands

- charge/on (start charging)
- charge/off (stop/pause charging)
- limitMode/{mode}
    - 0 = maxLimit
    - 1 = manually with App
    - 2 = Autopilot (Input (All) is used)
- limit/{limit}
    - charging limit between minLimit and maxLimit (kW)
    - changes only take every 15min affect (BMW only)

BMW Wallbox specific

Status:

- profiles (String, “|”-separated)
    - profiles must be set up on the BMW Wallbox
    - separate statistics (energy) are supported for each profile
- currentProfile (0,1)

Commands:

- profile/{profile}
    - switches the profile (0,1)

Central Objects

available since Miniserver 9.0

Covered Config Items

- CentralAlarm
- CentralAudioZone
- CentralGate
- CentralJalousie
- CentralLightController

Details

- controls - an array of JSON-Objects
    - uuid - uuid of the control
    - id - ID for selectedControls Command

Commands

- Command for selective control of objects
    - selectedcontrols/{ids - separated with comma}/{command}
    - If a selected control requires a visu password, it needs to be provided when sending the command to the Central Object.
    - If a control isn't accessible to a certain user, it also can't be controlled it via the Central Object.
    - for possible values for {command}, please see the concrete types below.
- CentralAlarm
    - on, off, quit, delayedon
- CentralAudio
    - play, pause
- CentralGate
    - open, close, stop
- CentralJalousie
    - FullUp, FullDown, shade, auto, NoAuto, stop
- CentralLightController
    - on, reset

ClimateController

available since Miniserver 10.0

Covered Config Items

- ClimateController

Details

- capabilities
    - Defined, what outputs are used
    - Possible values:
        - 0 = None
        - 1 = Only heating
        - 2 = Only cooling
        - 3 = Heating and cooling

Commands

- resetMaintenance
    - Resets the maintenance counter
- setServiceMode/{active}
    - Activates or deactivates the service mode
    - active:
        - Please check “serviceMode”
- ventilation/{active}
    - Activates the ventilator
- autoMode/{mode}
    - Activated the automatic mode
    - mode
        - 0 = Heating and cooling
        - 1 = heating
        - 2 = cooling
- setHeatingBoundary/{temp}
- setCoolingBoundary/{temp}

States

- controls
    - List of controls (IRCv2) added to the ClimateController
    - Possible values:
        - uuid = The uuidAction of the represented IRCv2
        - demand = The demand of the represented IRCv2
            - -1 = Cooling
            - 0 = None
            - 1 = Heating
    - Example:

        - [
            - {
                - uuid: "",
                - demand: -1, 0, 1
            - }
        -]
- currentMode
    - The current active mode
    - Possible values:
        - 0 = No requirement
        - 1 = Heating
        - 2 = Cooling
        - 3 = Heating boost
        - 4 = Cooling boost
        - 5 = Service mode
        - 6 = External Heater
- autoMode
    - 0 = Heating and cooling
    - 1 = Heating
    - 2 = Cooling
- currentAutomatic
    - The current active automatic mode
    - Possible values:
        - 0 = Automatic like conditions
        - 1 = Automatik like average temperature
- temperatureBoundaryInfo
    - Information about the temperature boundaries
    - Possible values
        - 0 = Not enough data
        - 1 = Ok
        - 2 = No data at all
- heatingTempBoundary
    - Temperature boundary for heating
- coolingTempBoundary
    - Temperature boundary for cooling
- actualOutdoorTemp
    - The outdoor temperature
    - -1000 = No temperature available
- overwriteReason
    - How the control is overwritten
    - Possible values:
        - 0 = Automatic
        - 1 = Boost
        - 2 = External Heater
        - 3 = Stop
- infoText

        - The name of the control connected to the currently active overwrite input
        - Possible values:
            - “...” = The name of the connected control
- serviceMode
    - What service mode setting is currently active
    - Possible values:
        - 0 = Off
        - 1 = Standby (Heating & Cooling OFF)
        - 2 = Heating On
        - 3 = Heating 1+2 On
        - 4 = Emergency Heating On
        - 5 = Cooling On
        - 6 = Colling 1+2 On
- nextMaintenance
    - Unix timestamp when the next maintenance must occur
- ventilation
    - State of Ventilation-Output

ColorPickerV2

This control type only appears as subcontrol of the LightControllerV2.

States

- color
  - The color as a string in the text property of the returned JSON
  - hsv(0,100,100) for RGB Values
  - temp(100,4483) for color temperatures (brightness, kelvin)
- sequence
  - an JSON-Object containing the description of a sequence
    - colors
      - An array of colors that will be iterated while being active
      - z.B.: [“hsv(0,100,100)”, ”hsv(100,100,100)”, “hsv(200,100,100)”]
    - interval
      - seconds how long it takes until one color changes to the next color.
      - between 60s and 3600s
- sequenceColorIdx
  - -1 if the sequence isn’t active
  - otherwise it’s the index of the active target color in the colors array of the current sequence.

Commands

- setFav/{favColorIdx}/{color}
  - Sets the favorite color
    - favColorIdx
      - The index of the color in the favoriteColors array
      - A new entry is created if the value already exists
      - Value is updated if it already exists
    - color
      - Either an HSV or an lumitech color string
        - hsv(0,100,100) for RGB Values
        - temp(100,4483) for Lumitech Values
      - favorite color is deleted if color is empty
- setFavSequence/{sequenceIdx}/{duration_seconds}/{colro_n...}
  - sequenceIdx
    - Index of Sequence array in GlobalStates
  - duration_seconds
    - duration in seconds

    - color_n / Maximal 6
        - Colors separated by /
- setSequence/{duration_seconds}/{color_n...}/{startIdx}
    - duration_second
        - duration in seconds
    - color_n/ Maximal 6
        - Color separated by /
    - startIdx
        - Start index of the sequence
        - -1 if the position in the sequence should not be modified.
- setBrightness/{value}
    - updates the sequencers brightness.
- hsv({hue},{saturation},{value})
    - sets a certain color, using HSV representation
- temp({brightness},{temperature})
    - sets a light with a certain color. (warm white, cold white)

ColorPicker

This control type only appears as subcontrol of the LightController.

Details

- pickerType
    - Rgb for RGB Pickers
    - Lumitech for Lumitech Pickers

States

- color
    - The color as a string in the text property of the returned JSON
    - hsv(0,100,100) for RGB Values
    - lumitech(100,4483) for Lumitech Values
- favorites
    - The favorites colors in the text property of the returned JSON
    - An array of either hsv or lumitech colors

Commands

- on
    - Enables the ColorPicker
- off
    - Disables the ColorPicker
- hsv(hue, sat, val)

    - set a new HSV color
- lumitech(brigthness,kelvin)
    - set a new color temperature, only supported for Lumitech Pickers
- setfav/{favIndex}/{color}
    - favIndex
        - The index of the favorit (1 - 4)
    - color
        - Either an HSV or an lumitech color string
            - hsv(0,100,100) for RGB Values
            - lumitech(100,4483) for Lumitech Values

Daytimer

Covered Config Items

- Timer/schedule

Details

- analog
    - indicates if the daytimer has an analog or digital output
- text (digital only)
    - on
        - the text if the value is 1
    - off
        - the text if the value is 0
- format (analog only)
    - the format of the value

States

- mode
    - current operating mode of the daytimer
- override
    - the remaining time of the time
- value
    - current value, 0 or 1 digital and a value for analog
- entriesAndDefaultValue
    - daytimer event with entries
    - a default value (used only for analog)
- resetActive
    - stays active as long as the reset input of the daytimer is active.

- Since Config 9.0

Commands

- pulse
  - activates the new value if an entry needs activation
- default
  - changes the default value in the analog daytimer
  - e.g.: default/8
- startOverride
  - starts the timer with a new value
  - e.g.: startOverride/{value}/{howLongInSecs}
- stopOverride
  - stops the timer
- set
  - change entries of the daytimer,
  - set/{numberOfEntries}/{entry}/{entry}/…,
  - {entry} = {mode};{fromMin};{toMin};{needsActivation};{valueOfEntry}
    - valueOfEntry will always be “0” in digital daytimers, or left out. Digital daytimers outputs are “On” as long as an entry exists.
    - from and to are to be given as minutes since midnight.
- modeslist
  - operating modes list sorted by the priority, on the end all weekdays from 3-9
  - 7,1,2,3,4,5,6,7,8,9

Intelligent Room Controller Daytimer v2

available since Miniserver 10.0

The daytimer used in the intelligent Room Controller v2 is an analog daytimer where the entries values identify the target temperature (identified by the temperature id, e.g. Comfort Temperature, Building Protection).

Commands

- set
  - set the calendar entries
- modeslist
  - operating modes list

Intelligent Room Controller Daytimer

The daytimers used in the intelligent Room Controller are analog daytimers where the entries values identify the target temperature (identified by the temperature id, e.g. Comfort Temperature, Empty House).

Commands

- setc
  - change entries of the cooling daytimer,
- set
  - for heating daytimer
- modeslistc
  - operating modes list for the cooling daytimer, modeslist for heating daytimer

Pool Daytimer

A pool daytimer is an analog daytimer where the analog value identifies what cycle (e.g. Backwash, Filter) is used while the entry is active.

Dimmer

Covered Config Items

- Dimmer

States

- position
  - The current position for the dimmer
- min
  - The current min value
- max
  - The current max value
- step
  - The current step value

Commands

- on
  - Sets the Dimmer to the last known position
- off
  - Disables the dimmer, sets position to 0
- {pos}
  - The position of the Dimmer
  - If position if over max, max will be set and if position is over min, min will be set

FanController

Details

- restrictedVentModes
    - Provides info on what vent modes are not supported by this control. If missing, it supports all modes.
    - e.g.:
        - “3” = at no time both the exhaust and the intake can be off.
        - “1,3” = it has no supply intake, only an exhaust.
- hasHeatExchanger
    - if the fan has a heat exchanger it can control.
    - 0 or missing = not controllable
    - 1 = analog controllable [0,100]
    - 2 = digital controllable
- hasHumidty
    - true if the controller makes use of the humidty
- hasOutdoorTemperature
    - true if the controller makes use of the outdoor temperature
- hasIndoorTemperature
    - true if the controller knows about the indoor temperature
- hasAirQuality
    - true if the controller makes use of an air quality sensor.

States

- mode
    - The active mode of the controller.
        - 0=off
        - 1=manual
        - 2=Autopilot (indoor temperature)
        - 3=dehumidifying
        - 4=Autopilot (air quality)
        - 5=night
        - 6=silent
        - 7=deactivated (Frost- or Heat-Protection)
        - 8=rush airing
- speed
    - 0-100% speed of the fan
- ventMode
    - 0 = supply and exhaust active
    - 1 = supply only active
    - 2 = exhaust only active

    - 3 = neither one is active.
- heatExchanger
    - 0-100% the heat exchanger is being used or not.
- remainingTime
    - -1 if a timer has run out, otherwise it's seconds until the current mode is being deactivated.
- totalTime
- targetTemp
- actualTemp
- actualOutdoorTemp
- actualHumidty
- actualAirQuality
- changeFilter
- winterSummerMode
    - 0= SummerMode
    - 1= WinterMode

Commands

- mode/{targetMode}/{optDuration}
    - activate a specific mode with an optional duration
    - targetMode → the mode that is to be activated, see states for possible values.
    - optDuration → duration in seconds
    - e.g..
        - mode/2
            - activate the autopilot (no duration)
        - mode/8/3600
            - activate the intensive airing for one hour
- mode/2/{duration}/{speed}/{ventMode}/{heatExchanger}
    - activate manual mode (with a duration)
    - duration → seconds until the ventilation stops
    - speed → set the fan speed
    - ventMode → see state ventMode.
    - heatExchanger → value depending on the speed
    - Allowed modes:
        - 0 = off
        - 1 = autopilot
        - 2 = manual
        - 3 = intensiv
        - 4 = silence

Fronius

Covered Config Items

- Energy Monitor

States

- prodCurr
    - kW current production power
- prodCurrDay
    - kWh energy production all over the current day
- prodCurrMonth
- prodCurrYear
- prodTotal
    - kWh energy production since setting up
- consCurr
    - kW current consumption power
- consCurrDay
    - kWh energy consumed throughout the current day
- gridCurr
    - kW current grid consumption/delivery power
    - if negative, power is being delivered to the grid
    - available since Miniserver 8.1
- batteryCurr
    - kW current battery charging/usage power.
    - if negative, the battery is charging
    - available since Miniserver 8.1
- stateOfCharge
    - 0-100, represents the charging state of the battery. 100 = fully charged.
    - available since Miniserver 8.1
- earningsDay
    - how much money was earned by either consuming the produced power yourself instead of consuming it from the grid, or by exporting unused produced power to the grid. Depends on priceDelivery and priceConsumption
- earningsMonth
- earningsYear
- earningsTotal
- priceDelivery
    - Price per unit when exporting to the grid
- priceConsumption
    - Price per Unit while consuming from the grid
- co2Factor

        - How much co2 does it take to produce one kWh, used to compute CO2 savings
- generatorType
    - 0 = Fronius
        - data supplied by a Fronius generator
    - 1 = Inputs
        - data supplied by the block inputs
    - 2 = Kostal
        - data supplied by a Kostal devices
- mode
    - the mode gives info on what data sources are available on this energy monitor. As there are several combinations, a bitmask is used to determine whether or not certain sources are available.
        - Bit 0 = Production data available
        - Bit 1 = Consumption data available
        - Bit 2 = Battery data available
            - available since Miniserver 8.1
    - Here are some examples for possible resulting values
        - 3 = Production and Consumption available
        - 1 = Production only
        - 2 = Consumption only
        - 0 = No data available
        - 7 = Production, Consumption and Battery available
        - ..
- online
    - 0 = online
    - 1 = offline

Gate

Covered Config Items

- Gate

Details

- animation
    - 0 = Garage Door
    - 1 = Single Gate opening to the left
    - 2 = Single Gate opening to the right
    - 3 = Gate opening to both sides

    - 4 = Folding door opening to the left
    - 5 = Folding door opening to the right

States

- position
    - the position from 1 = up and 0 = down
- active
    - -1 = close
    - 0 = not moving
    - 1 = open
- preventOpen
    - 0 = not preventing opening of door
    - 1 = preventing opening of door
- preventClose
    - 0 = not preventing closing of door
    - 1 = preventing closing of door

Commands

- open
    - Opens the Gate
- close
    - Closes the Gate

Heatmixer

Covered Config Items

- Mixing Valve Controller

States

- tempTarget
    - temperature the controller currently aims for
- tempActual
    - actual temperature reported by the sensor attached to the input

Hourcounter

Covered Config Items

- Maintenance counter

States

- total
  - total number of seconds the counter has been active so far
- remaining
  - how many seconds left until the next maintenance is required
  - 0 if required or overdue
- lastActivation
  - the timestamp (in seconds) when the counter was activated the last time.
  - updated on each new activation
- overdue
  - 0 if not overdue, otherwise maintenance is required
- maintenanceInterval
  - seconds until the next maintenance
- stateUnit
  - desired output unit on the UI (state values remain in seconds!)
    - 0 = seconds
    - 1 = minutes
    - 2 = hours
    - 3 = days
- active
  - 0/1, whether or not the counter is currently active
- overdueSince
  - seconds since the maintainanceInterval was exceeded
  - 0 maintenance is not required yet.

Commands

- reset
  - will cause a reset of the following values
    - remaining to maintenanceInterval
    - overdue to 0
    - overdueSince to 0
- resetAll
  - like reset, but also sets

        - total to 0
        - lastActivation to 0

InfoOnlyAnalog

Covered Config Items

- Virtual state

Details

- format
    - the format of the value

States

- value
    - the current value of the virtual state

InfoOnlyDigital

Covered Config Items

- Virtual state

Details

- text
    - on
        - on text if the value is 1
    - off
        - off text if the value is 0
- image
    - on
        - uuid of the “on”-image
    - off
        - uuid of the “off”-image
- color
    - on
        - text color if the value is 1
    - off

        - text color if the value is 0

States

- value
    - the current value of the virtual state

Intelligent Room Controller v2

available since Miniserver 10.0

Covered Config Items

- Intelligent room controller V2

Details

- **format**
    - defines Temperature-Format (°C or °F)
- **timerModes**
    - Defined the available timers
    - A manual mode is also available (id = 3), but not available in this structure
    - Structure description
        - name = Name of the mode
        - id = ID of the mode, defined by the Miniserver (is used in **states.activeMode**)

States

- activeMode
    - The active mode
    - Available modes:
        - 0 = Economy
        - 1 = Comfort temperature
        - 2 = Building protection
        - 3 = Manuel
- operatingMode
    - Available modes
        - 0 = Automatic, heating and cooling allowed
        - 1 = Automatic, only heating allowed
        - 2 = Automatic, only cooling allowed
        - 3 = Manuel, heating and cooling allowed

        - 4 = Manuel, only heating allowed
        - 5 = Manuel, only cooling allowed
- overrideEntries
    - Shows an array of the current active override or an empty array if no active override exists
    - Structure
        - start = Seconds since 2009
        - end = 0 if currently active or seconds since 2009
        - rason
        - 0 = None
        - 1 = Someone is present -> *Comfort mode is active*
        - 2 = Window open -> *Eco+ mode is active*
        - 3 = Comfort overrid
        - 4 = Eco override
        - 5 = Eco+ override
        - 6 = Prepare State Heat Up
        - 7 = Prepare State Cool Down
    - source
        - the name of the source, e.g: The name of the connected input of the block or null it not available
    - isTimer
        - Indicates, that the timer has been started via the app
        - true | false
- prepareState
    - Possible Values
        - -1 = Cooling down
        - 0 = No Action
        - 1 = Heating up
- overrideReason
    - PossibleValues
        - Please refer to **overrideEntries.reason**
- tempActual
    - The current Temperature
- tempTarget
    - The current target Temperature
- comfortTemperature
    - Literally the comfort temperature
- comfortTolerance
    - The allowed comfort tolerance (±)
- absetMinOffset

    - Literally the minimal temperature offset of the economy mode
- absentMaxOffset
    - Literally the maximal temperature offset of the economy mode
- frostProtectTemperature
    - Literally the minimal temperature of the Building protection
- heatProtectTemperature
    - Literally the maximal temperature of the Building protection
- comfortTemperatureOffset
    - The offset of the comfort temperature

Commands

- override/{modeId}/[{until}]/[{temp}]
    - Starts a override timer
    - modeId
        - The requested modeId of the timer
    - [until]
        - Seconds since 2009 when the timer should end
    - [temp]
        - Only needed if we specifically want to set the temperature (manual mode)
- stopOverride
    - Stops a currently running override timer
- setComfortTemperature/{temp}
    - temp
        - The comfort temperature
- setComfortTolerance/{tolerance}
    - tolerance
        - The tolerance
        - min
            - 0.5
        - max
            - 3.0
- setAbsentMinTemperature/{offset}
    - offset
        - The minimum temperature in Eco-Mode
- setAbsentMaxTemperature/{offset}
    - offset
        - The maximum temperature in Eco-Mode
- setManualTemperature/{temp}
    - temp

        - The Manual Target-Temperature
- setOperatingMode/{opMode}
    - opMode
        - Please check **states.operatingMode**
- setComfortModeTemp/{temp}
    - temp
        - Temperature described as an numerical value
    - *How to calculate the temp:*
        - *states.comfortTemperatur + states.comfortTemperatureOffset + step(0.5)*
        - *states.comfortTemperatur + states.comfortTemperatureOffset - step(0.5)*
- set/
    - Set calendar-Entries
- modeslist/
    - Set new Modeslist for Daytimer

Sub-Controls

Intelligent Room Controller

Covered Config Items

- Intelligent room controller

Details

- restrictedToMode
    - 0
        - Visualize heating and cooling
    - 1
        - Visualize cooling only
    - 2
        - Visualize heating only
- heatPeriodStart, heatPeriodEnd
    - Provided if this room controller is using a custom heating period. Returns the month and day when the heating period of this IRoomController will start and end. Missing if the global heating period is used.
    - Modified in Config 8.3
- coolPeriodStart, coolPeriodEnd
    - the same as heatPeriodStart/heatPeriodEnd but for the cooling period.
- temperatures

    - id’s of the temperature modes to identify values from the states, e.g.
    - temperature-ids
        - 0 = Economy
        - 1 = Comfort Heating
        - 2 = Comfort Cooling
        - 3 = Empty House
        - 4 = Heat Protection
        - 5 = Increased Heat
        - 6 = Party
        - 7 = Manual
    - isAbsolute
        - is the value depending of comfort heating/cooling or an absolute value

States

- tempTarget
    - the current target temperature
- tempActual
    - the current temperature
- error
    - could be a big difference between target and actual temperature, the actual temperature is bigger than the heat protection temperature or the actual temperature is lower than the empty house temperature
- mode
    - information about the mode of the IRoomController
    - modes:
        - 0 = Autopilot
            - cooling or heating depending on the heating/cooling period
            - only returned if neither a heating nor a cooling period is active
        - 1 = Autopilot (currently heating),
        - 2 = Autopilot (currently cooling),
        - 3 = Autopilot heating,
        - 4 = Autopilot cooling,
        - 5 = manual heating,
        - 6 = manual cooling
- serviceMode
    - the current service mode index
    - serviceModes:
        - 0 = off
        - 1 = heating and cooling off
        - 2 = heating on cooling of
        - 3 = heating off cooling on

        - 4 = heating and cooling on
- currHeatTempIx
    - the current heating temperature index of the temperatures
- currCoolTempIx
    - the current cooling temperature index of the temperatures
- override
    - the remaining time of the timer
- openWindow
    - if the window is currently opened
- overrideTotal
    - the total time with which the timer was started
- manualMode
    - if the user overrides with manual intervention
    - modes:
        - 0 = off
        - 1 = comfort overriding
        - 2 = economy overriding
        - 3 = timer overriding (through app)
        - 4 = movement/presence
- temperatures
    - an array of temperatures, index is the same as the id of the existing temperatures in the details object
- stop
    - While this state is on, all outputs of the room controller will remain off, regardless of the temperatures. The rest of the room controller will respond as usual.
    - available since Miniserver 9.0

Commands

- mode
    - in which mode the IRoomController should work (0-6)
        - Autopilot (currently cooling or heating, nr 1 & 2) are not manually selectable, they are chosen depending on the heating/cooling period. Use 3 & 4 instead.
- service
    - to activate the service mode with an id from 0 - 4
- starttimer
    - starts the timer with a temperature id and remaining seconds
- stoptimer
    - stops the timer
- settemp

- changes the value of an temperature with a temperature id and the new value

Sub-Controls

- 2 Intelligent room controller daytimer for heating and cooling

Intercom

Covered Config Items

- Door Controller

Details

- deviceType
  - 0 = Custom/Unknown
  - 1 = Loxone Intercom
  - 2 = Loxone Intercom XL
  - available since Miniserver 8.0
- videoInfo
  - An empty object for legacy reasons
- audioInfo
  - An empty object for legacy reasons
- The content of videoInfo and audioInfo have been moved to securedDetails in Version 8.1
- lastBellEventImages
  - true if the Miniserver does store images for the last bell event entries.
  - false or missing if it does not.
  - available since Miniserver 8.3

Secured Details

- videoInfo
  - alertImage
    - an (optional) path to a still image (jpg), used to save pictures for “lastBellEvents”
  - streamUrl
    - The path to either a mjpg stream or jpg images
  - user
  - pass
- audioInfo
  - host

        - the host portion for making a sip-call
    - user
        - the (optional) user portion for making an sip-call
    - pass
        - Loxone Intercoms only - the password to authenticate the call
        - available since Miniserver 8.0

States

- bell
    - 0 = not ringing.
    - 1 = ringing.
- lastBellEvents
    - Text containing the timestamps for each bell-activity that wasn't answered.
    - YYYYMMDDHHMMSS, separated by a pipe
    - e.g.: "20151001074904|20151001075008|20151008171552"
    - recorded images (if available) can be accessed via "camimage/{*uuidAction*}/{*timestamp*}"
- version
    - Loxone Intercoms only - text containing the currently installed firmware versions.

Commands

- answer
    - Will deactivate the bell.

Sub-Controls

- 0-3 Outputs of type Pushbutton
    - since 8.0 outputs can handle a 'pulse' command. It was ignored in earlier versions.

Jalousie

Covered Config Items

- Blinds
- Automatic blinds
- Automatic blinds integrated

Details

- isAutomatic
  - If this is an Jalousie with Autopilot (automatic blinds, automatic blinds integrated)
- animation
  - The animation type of the JalousieControl
    - 0 = Blinds
    - 1 = Shutters
    - 2 = Curtain both sides
    - 3 = Not supported
    - 4 = Curtain Left
    - 5 = Curtain Right
- availableModes
  - ConfigMode, not documented.

States

- up
  - Jalousie is moving up
- down
  - Jalousie is moving down
- position
  - The position of the Jalousie, a number from 0 to 1
    - Jalousie upper position = 0
    - Jalousie lower position = 1
- shadePosition
  - The orientation of the slats (important only for animation 0 = blinds), a number from 0 to 1
    - horizontal slats = 0
    - vertical slats (fully shaded) = 1
- safetyActive
  - Only used by ones with Autopilot, this represents the safety shutdown
- autoAllowed
  - Only used by ones with Autopilot
- autoActive
  - Only used by ones with Autopilot
- locked
  - Only used by ones with Autopilot, this represents the output QI in Loxone Config. Overrules the safetyActive, as not even for moving to the safety position is allowed.
- hasEndposition

    - ConfigMode, not documented.
- mode
    - ConfigMode, not documented.
- learningStep
    - ConfigMode, not documented.
- infoText
    - informs e.g. on what caused the locked state, or what did cause the safety to become active.
    - available since Miniserver 9.0

Commands

- up
    - Sends an Up command to the Jalousie, as long as no UpOff or other direction command has been sent the Jalousie will go UP
- UpOff
    - Sends an UpOff command to the Jalousie, this will stop the up motion of the Jalousie
- down
    - Sends an Down command to the Jalousie, as long as no DownOff or other direction command has been sent the Jalousie will go Down
- DownOff
    - Sends an DownOff command to the Jalousie, this will stop the down motion of the Jalousie
- FullUp
    - Triggers a full up motion
- FullDown
    - Triggers a full down motion
- shade
    - Shades the Jalousie to the perfect position
- auto
    - Enables the Autopilot of the Jalousie (if the control supports it)
- NoAuto
    - This disables the Autopilot mode for the Jalousie (if the control supports it)

NFC Code Touch

Covered Config Items

- NFC Code Touch

Details

- accessOutputs

- object with output names, q1-q6
- if output isn't used, it isn't listed

States

- historyDate
  - unix timestamp in milliseconds from the latest history entry
  - is 0 if no history exists
  - reload the history if this updates
- codeDate
  - unix timestamp in milliseconds from the latest change of some code
  - the codes has changed, reload them

Commands

- output/{outputNr}
  - outputNr can be 1-6
  - sends an impuls to the specific output
  - Code 423
    - user is not permitted at the moment
- history
  - returns JSON array with history entry objects
  - entry structure:
    - ts
      - unix timestamp in utc
    - output
      - int, 1-6
    - type
      - int, 0-6
      - 0 = code (can be associated with one user)
      - 1 = code (with name)
      - 2 = code (ambiguous, used from multiple users)
      - 3 = nfc (can be associated with one user)
      - 4 = nfc (with name)
      - 5 = nfc (ambiguous, used from multiple users)
      - 6 = via app
      - 7 = denied code
      - 8 = denied NFC Tag
    - user
      - string, optional, name of user (if type is 0, 3 or 6)
    - description
      - string optional
        - if type 1: name of code

        - if type 3 or 4: name of tag
- codes
    - JSON array with Code objects from this keypad
    - Code structure:
        - uuid
            - string
        - name
            - string
        - isActive
            - boolean, if code is currently active
        - type
            - int, 0-2
            - 0 = permanent code
            - 1 = one-time code
            - 2 = time-dependent code
        - outputs
            - permitted outputs this code can be used for
            - bitmask
                - 1 = q1, 2 = q2, 3 = q1+q2, 4 = q3, 5 = q1+q3, 6 = q2+q3...
        - standardOutput
            - int, 1-6
        - timeFrom
            - optional, if type is 2
            - unix timestamp in utc
        - timeTo
            - optional, if type is 2
            - unix timestamp in utc
- code/create/{name}/{code}/{type}/{outputs}/{standardOutput}
    - creates a new code with the following properties:
    - name
        - string, URIComponent encoded
    - code
        - string (numeric)
    - type
        - int, 0-2 (see Code structure above)
    - outputs
        - bitmask (see Code structure above)
    - standardOutput
        - int, 1-6
- code/create/{name}/{code}/{type}/{outputs}/{standardOutput}/{timeFrom}/{timeTo}
    - same as previous command but for codes with type 2:

    - timeFrom
        - unix timestamp in utc
    - timeTo
        - unix timestamp in utc
- code/update/{uuid}/{isActive}/{name}/{type}/{outputs}/{standardOutput}
    - updates the code with the following properties:
    - uuid
        - string, code uuid
    - isActive
        - true/false, if the code should be activated or deactivated
    - code
        - code or -1 if code does not change
- code/update/{uuid}/{isActive}/{name}/{type}/{outputs}/{standardOutput}/{timeFrom}/{timeTo}
    - updates the code with the following properties:
    - timeFrom
    - timeTo
- code/activate/{uuid}
    - activates code
- code/deactivate/{uuid}
    - deactivates code
- code/delete/{uuid}
    - deletes the code

LightController

Covered Config Items

- Lighting controller
- Hotel lighting controller

Details

- movementScene
    - The scene number that is assigned as the movement scene

States

- activeScene
    - The current active scene number
- sceneList
    - Returns a JSON-Object in the following format
    - uuid -> the UUID of the LightControl
    - uuidIcon -> This is only used by the Status Control

    - text -> This is an array of Scenes separated by a “,”
        - "1=\"Szene1\",2=\"Szene2\",7=\"Szene7\""

Commands

- off
    - Enables lightscene 0 (All off)
- on
    - Enables lightscene 9 (All on)
- {sceneNumber}
    - This will activate the scene
- {sceneNumber}/learn/{sceneName}
    - This learns the current output values to the given scene and overrides it if the scene already exists. Also used to rename a scene or create a new one.
- {sceneNumber}/delete
    - This deletes the given scene
- plus
    - Changes to the next scene
    - available since Miniserver 8.1
- minus
    - Changes to the previous scene
    - available since Miniserver 8.1

LightControllerV2

Covered Config Items

- Lighting controller V2

Details

- masterValue
    - the UUID of the master brightness in the subcontrols
    - will always be present.
- masterColor
    - the UUID of the master rgb in the subcontrols
    - will only be present if there are subcontrols of the type colorPicker

States

- activeMoods
    - A list of mood ids that are currently active (JSON-Array)
- moodList

    - Returns an array with JSON-Objects each containing the attributes listed below. The position in the array reflects the list order.
    - name
        - The userfriendly name for this mood.
    - id
        - An ID that uniquely identifies this mood (e.g. inside activeMoods) (e.g. ID1, ID2,..)
    - t5
        - whether or not this mood can be controlled with a t5 input
    - static
        - If a mood is marked as static it cannot be deleted or modified in any way. But it can be moved within and between favorite and additional lists.
    - used
        - Bitmask that tells if the mood is used for a specific purpose in the logic. If it's not used, it can be removed without affecting the logic on the Miniserver.
            - 0: not used
            - 1: this mood is activated by a movement event
            - 2: a T5 or other inputs activate/deactivate this mood
            - 4: this mood is activated by the Alarm Clock
- favoriteMoods
    - [ID1,ID12,ID4,ID5]
- additionalMoods
    - [ID2,ID3,ID6,ID7,ID8,ID9,ID10,ID11,ID13]

Commands

- changeTo/{moodId}
    - This will activate the mood, other moods will be deactivated
- addMood/{moodId}
    - adds the provided mood and mixes it with the other moods that are active
- removeMood/{moodId}
    - deactivates the provided mood, leaves the other moods untouched.
- moveFavoriteMood/{moodId}/{newIdx}
    - moves a mood within the favorite mood list
    - moodId
        - Id of the mode
    - newIdx
        - New Index of the mood
- moveAdditionalMood/{moodId}/{newIdx}
    - Moves a mood within the additional mood list

    - moodId
        - Id of the mode
    - newIdx
        - New Index of the mood
- moveMood/{moodId}/{newIdx}
    - Moves a mood within the mood list
    - moodId
        - id of the mode
    - newIdx
        - New Index of the mood
- addToFavoriteMood/{moodId}
    - Moves a mood to the favorite mood (last index)
    - moodId
        - Id of the mode
- removeFromFavoriteMood/{moodId}
    - Moves a mood from the favorite mood list to the additional mood list
    - moodId
        - Id of the mode
- learn/{moodId}/{moodName}
    - This learns the current output values to the given mood and overrides it if the mood already exists. Also used to rename a mood or create a new one.
    - moodIds < 8 are linked to the appropriate T5 Input
- delete/{moodId}
    - This deletes the given mood.
- plus
    - Changes to the next mood
- minus
    - Changes to the previous mood

LightsceneRGB

Covered Config Items

- RGB lighting controller

Details

- sceneList
    - An array of scene names

States

- activeScene
    - The current active scene number

- color
    - A string of the current color
        - hsv(0, 100, 100)

Commands

- off
    - Enables lightscene 0 (All off)
- on
    - All on
- {sceneNumber}
    - this will activate the scene
- {sceneNumber}/learn
    - this will override the selected scene with the new selected color If this scene does not exist no new scene will be created!

Meter

Covered Config Items

- Utility meter

Details

- actualFormat
    - format specifier for the value of “actual”
- totalFormat
    - format specifier for the value of “total”

States

- activeOutput
    - ID of the currently active output.
    - 0 means no output is selected, show “All Off” (if available)

Commands

- reset
    - Deselects the currently

PoolController

Covered Config Items

- Pool Controller

Info

The PoolControl only works with an Aquastar Air Valve.

Details

- valveType
  - The types of the valve
    - 0 = No Valve
    - 1 = Aquastar
  - hasEco
    - If something is connected to the ECO input
  - hasTargetTemp
    - Is input T connected
  - hasActualTemp
    - Is input AI connected
  - hasWaterLevel
    - Is input WI connected
  - waterLevelUnit
    - The unit for the waterlevel
  - hasCoverPosition
    - Is input CP connected
  - hasCover
    - Is output Qco and Qcc connected
  - swimmingMachineType
    - Is dependent on output AQsm
      - 0 = no swimming machine
      - 1 = digital value
      - 2 = analog value
  - hasCustom1
    - Is input AI1 connected
  - hasCustom2
    - Is input AI2 connected
  - customName1
    - Name for value of input AI1

    - customName2
        - Name for value of input AI2
    - customUnit1
        - Unit for value of AI1
    - customUnit2
        - Unit for value of AI2
    - filterBounds
        - Min and max values for mode “Filter” in seconds
    - backwashBounds
        - Min and Max values for mode “backwash” in seconds
    - rinseBounds
        - Min and Max values for mode “rinse” in seconds
    - circularBounds
        - Min and Max values for mode “circulate” in seconds
    - drainBounds
        - Min and Max values for mode “drain” in seconds
    - hasHeating
        - Is output Qh connected
    - hasCooling
        - Is output Qc connected

States

- currentOpMode
    - current operating mode
        - 0 = Out of order
            - while the reset input is active
        - 1 = Autopilot
        - 2 = Servicemode
- currentTempMode
    - current temperature mode
        - 0 = off
        - 1 = Autopilot
        - 2 = Autopilot heating
        - 3 = Autopilot cooling
        - 4 = manual heating
        - 5 = manual cooling
- tempActual
    - The actual water temperature
- tempModeCycleActive
    - If the temperature regulation cycle is active
- tempTarget

    - The target temperature
- waterLevel
    - The actual water level
- custom1
    - The value of AI1
- custom2
    - The value of AI2
- heatingApproved
    - If heating is approved, only used if hasHeating is true
- coolingApproved
    - If cooling is approved, only used if hasCooling is true
- ecoActive
    - If the eco mode is active
- swimmingMachine
    - This value can either be digital or analog, please refer to "swimmingMachineType" in the control details
- coverPosition
    - Analog value of the cover position
        - 0.0 = open
        - 1.0 = closed
- coverOpening
    - If the cover is opening right now
- coverClosing
    - If the cover is closing right now
- currentCycle
    - Current active cycle
        - 0 = No cycle is active
        - 1 = Filter
        - 2 = Flushing
        - 3 = Circulate
        - 4 = Drain
- remainingTime
    - Remaining time of the active cycle in seconds
- valvePosition
    - The current position of the valve
        - -1 = Valve moves
        - 0 = Filter
        - 1 = Backwash
        - 2 = Clearwash
        - 3 = Circulate
        - 4 = Closed
        - 5 = Drain

                - 2-way ball valve is activated
                - Valve moves to position drain
                - Activates pump for a given time
                - After given time valve will be deactivated, but the 2-way ball valve and the valve stays at the drain position
            - 6 = Relieve
                - This is used to relieve the valve in the winter
- pump
    - If the pump is active
- drainValve
    - If the drainvalve is opened. There might be a separate drainValve attached that needs to be opened besides setting the valvePosition to drain.
- delayTime
    - The time of the delay
        - This must be in the range of “delayBounds”
- filterTime
    - The time in seconds the mode “Filter” will be active
        - This must be in the range of “filterBounds”
- backwashTime
    - The time in seconds the mode “BackwashTime” will be active
        - This must be in the range of “backwashBounds”
- rinseTime
    - The time in seconds the mode “Rinse” will be active
        - This must be in the range of “rinseBounds”
- circulateTime
    - The time in seconds the mode “Circulate” will be active
        - This must be in the range of “circulateBounds”
- drainTime
    - The time in seconds the mode “Drain” will be active
- error
    - Error status of the PoolController (available since Miniserver 8.0)
        - 0 = No error is or was present
        - 1 = An error was present
        - 2 = An error is currently present

Commands

- coverClose
    - Closes the cover if one is connected
- coverOpen
    - Opens the cover if one is connected
- operatingMode/{opMode}

    - Activates the given operating mode
        - 0 = Out of order
        - 1 = Autopilot
        - 2 = Servicemode
- tempMode/{tempMode}
    - Activates the given temperature mode
        - 0 = off
        - 1 = Autopilot
        - 2 = Autopilot heating
        - 3 = Autopilot cooling
        - 4 = manual heating
        - 5 = manual cooling
- eco/{state}
    - state
        - 0 = off
        - 1 = on
- targetTemp/{temperatur}
    - Sets the given target temperature
- swimmingMachine/{value}
    - value can be either analog (0.0 - 1.0) or digital. Please refer to “swimmingMachineType”
- startCycle/{cycleID}/{seconds1}/{seconds2}
    - starts the given cycle with the given seconds
        - cycleID”
            - 1 = Filter
            - 2 = Flushing
            - 3 = Circulate
            - 4 = Drain
        - seconds1 = The duration the cycle will be active
            - If no seconds are given the values from the states (filterTime, circleTime,..) will be used
- filter
    - Short for “startCycle/1”
- backwash
    - short for “startCycle/2”
- circulate
    - short for “startCycle/3”
- drain
    - short for “startCycle/4”
- valvePos/{position}
    - Sets the valve position
        - position

                - 0 = Filter
                - 1 = Backwash
                - 2 = Clearwash
                - 3 = Circulate
                - 4 = Closed
                - 5 = Drain
            - Position 6 = Relieve-position cannot be set using a command
- pump/{state}
    - Activates or deactivates the pump
    - state
        - 0 = off
        - 1 = on
- drainValve/{state}
    - Opens or closes the drainvalve
    - state
        - 0 = open
        - 1 = close
- reset
    - Pulse for reset
    - reset/1 will activate “Out of order”
- disable/{state}
    - Disables or enables childlock
    - state
        - 0 = off
        - 1 = on
- delayTime/{time}
    - Sets the delaytime in seconds
    - NOTE: This value must be between “delayBounds”
- filterTime/{time}
    - Sets the filtertime in seconds
    - NOTE: This value must be between “filterBounds”
- backwashTime/{time}
    - Sets the backwashtime in seconds
    - NOTE: This value must be between “backwashBounds”
- rinseTime/{time}
    - Sets the rinseTime in seconds
    - NOTE: This value must be between “rinseBounds”
- circulateTime/{time}
    - Sets the circulateTime in seconds
    - NOTE: This value must be between “circulateBounds”
- drainTime/{time}
    - Sets the drainTime in seconds

    - NOTE: This value must be between “drainTime”
- approveHeating/{value}
    - Approves or disapproves heating
    - value
        - 0 = approve
        - 1 = disapprove
- approveCooling/{value}
    - Approves or disapproves heating
    - value
        - 0 = approve
        - 1 = disapprove
- skipDelay
    - This cancels the delay
- ackError
    - Acknowledges the current error (available since Miniserver 8.0)

Pushbutton

Covered Config Items

- Virtual Input (Pushbutton)

States

- active
    - the current state of the pushbutton

Commands

- pulse
    - if the button was only tapped for a very short time, use pulse instead of on and off
- on
    - when the button is hit but not released immediately
- off
    - to deactivate the button after an on-command

Radio

Covered Config Items

- Radio buttons (8x)
- Radio buttons (16x)

Details

- allOff
  - the name shown when no output is selected
  - Empty string if selecting neither one of the outputs is not an option
- outputs
  - set of output names with their ID as key. (1 - 16)
  - there might be missing IDs (e.g.: 1,2,5,8 is a valid set of IDs)

States

- activeOutput
  - ID of the currently active output.
  - 0 means no output is selected, show “All Off” (if available)

Commands

- reset
  - Deselects the currently selected ID, changes activeOutput to 0
- *ID*
  - Simply sending an ID will activate the corresponding output & change activeOutput
  - 0 cannot be selected directly, only via ‘reset’

Remote

Covered Config Items

- Media controller

Details

- favoritePad
  - 0 = D-Pad should be initially visible
  - 1 = Number-Pad should be initially visible
- modeList
  - Object with the modes (usedButtons are currently not used!)
    - { “1”: { “name”: “Mode One”, “usedButtons”: []}}

States

- timeout
  - The timeout in milliseconds
- mode
  - The key for the current mode (The key is for the modeList)
  - 0 means “no mode selected” = all off.
- active
  - Will be true if the Miniserver is sending the commands for switching the modes or power on
  - since Config 8.0

Commands

- mode/{modeID}
  - This enables the mode with the given ID
  - cannot select mode “0” - use reset for that.
- on
  - This enables the AQp (Power) output
- off
  - This disables the AQp (Power) output
- mute
  - Represents the Mute Button
- play
  - Represents the Play Button
- pause
  - Represents the Pause Button
- stop
  - Represents the Stop Button
- rewind
  - Represents the Rewind Button
- previous
  - Represents the Previous Button
- next
  - Represents the Next Button
- forward
  - Represents the Forward Button
- menu
  - Represents the Menu Button
- info
  - Represents the Info Button
- exit

    - Represents the Exit Button
- guide
    - Represents the Guide Button
- volplus
    - Represents the press of the Volume Plus Button
- volminus
    - Represents the press of the Volume Minus Button
- volplusoff
    - Represents the release of the Volume Plus Button
- volminusoff
    - Represents the release of the Volume Minus Button
- prgplus
    - Represents the press of the Program Plus Button
- prgminus
    - Represents the press of the Program Minus Button
- prgplusoff
    - Represents the release of the Program Plus Button
- prgminusoff
    - Represents the release of the Program Minus Button
- return
    - Represents the Return Button
- btnred
    - Represents the Red Button
- btnblue
    - Represents the Blue Button
- btnyellow
    - Represents the Yellow Button
- btngreen
    - Represents the Green Button
- dirok
    - Represents the D-Pad OK Button
- dirup
    - Represents the press of the D-Pad Up Button
- dirupoff
    - Represents the release of the D-Pad Up Button
- dirdown
    - Represents the press of the D-Pad Down Button
- dirdownoff
    - Represents the release of the D-Pad Down Button
- dirleft
    - Represents the press of the D-Pad Left Button
- dirleftoff

    - Represents the release of the D-Pad Left Button
- dirright
    - Represents the press of the D-Pad Right Button
- dirrightoff
    - Represents the release of the D-Pad Right Button
- num{x}
    - This sends the number “x”, “x” goes from 0-9
        - Example: “num1” for number 1
- number/{x}
    - This sends the number “x”, “x” can be any positive number
        - Example: “number/18” for number 18
- reset
    - turns off all devices of the current mode & changes to mode 0 (= no mode active)
    - available since Miniserver 8.0

Sauna

Covered Config Items

- Sauna controller
- Sauna controller with evaporator

Details

- hasVaporizer
    - determines whether or not it is a full featured sauna with vaporizer
- hasDoorSensor
    - whether or not the value of the door sensor can be visualized (might not be attached)

States

- active
    - whether or not the sauna is active (!= power!)
- power
    - is it currently heating up
- tempActual
    - the actual temperature inside the sauna
- tempBench
    - the actual temperature provided by the bench sensor
- tempTarget
    - the current target temperature
- fan

    - is the fan on
    - indicates the airing phase (if drying is on too)
    - the airing phase will stop after the airing-time configured on the block is reached.
- drying
    - indicates that the “drying phase” is on
    - if the fan is on too, it’s called the “airing phase”
    - the drying will stop once the targetTemp is reached
- doorClosed
    - active if door is closed, only to be used if hasDoorSensor is true
- presence
    - forwards the state of the presence input of the block
- error
    - digital indicator for a sauna error
- saunaError
    - indicates what error has occurred and why the sauna has shut down.
    - 0 = no error
    - 1 = too hot
- timer
    - seconds left of the sauna timer
- timerTotal
    - total number of seconds of the sauna timer
- lessWater (evaporator only)
    - becomes active if the evaporator runs out of “water”
- humidityActual (evaporator only)
    - actual humidity inside the sauna
- humidityTarget (evaporator only)
    - target humidity inside the sauna
- mode (evaporator only)
    - when an evaporator is present, different sauna modes can be used, these are identified by a modeNr
    - 0 = Off
    - 1 = Finnish manual operation
    - 2 = Humidity manual operation
    - 3 = Finnish automatic operation (80°C)
    - 4 = Herbal automatic sauna (45°C, 50%)
    - 5 = Soft steam bath automatic (50°C, 50%)
    - 6 = Warm air bath automatic (45°C, 45%)

Commands

- on

    - turns the sauna on
- off
    - turns the sauna off right away (no airing/drying phase)
- fanoff
    - turns the fan off
- fanon
    - turns the fan off, only works if sauna is active
- temp/{target}
    - set the target temperature (for manual mode)
- humidity/{target}
    - set the target humidity (hasVaporizer only)
- mode/{modeNr}
- pulse
    - changes between the sauna activity-states
        - off -> on
        - on -> drying
        - drying -> airing
        - airing -> off
- starttimer
    - starts the sand timer, will count down from timerTotal

Slider

Covered Config Items

- Virtual Input (Slider)

Details

- format
    - the format of the value
- min
    - the minimum value
- max
    - the maximum value
- step
    - the step to the next value of the slider when pressing “-” or “+”

States

- value
    - the current value of the slider

- error
    - indicates an invalid value of the slider

Commands

- {number}
    - value for the slider
    - between min and max

SmokeAlarm

Covered Config Items

- Fire/water alarm

Details

- hasAcousticAlarm
    - returns true if the smoke alarm control has an acoustic alarm configured
- availableAlarms
    - Bitmask showing what is being monitored (may be a combination)
        - 0x01: Smoke
        - 0x02: Water
        - 0x04: Temperature
        - 0x08: Arc Fault (electrical wiring), available since Miniserver 9.3

States

- nextLevel
    - the ID of the next alarm level
        - 1 = Silent
        - 2 = Acustic
        - 3 = Optical
        - 4 = Internal
        - 5 = External
        - 6 = Remote
- nextLevelDelay
    - The delay of the next level in seconds, this can be specified with the parameters D1 - D6 in Loxone Config. This increments every second...
- nextLevelDelayTotal
    - The total delay of the next level in seconds, this can be specified with the parameters D1 - D6 in Loxone Config.

- level
  - The ID of the current alarm level
    - 1 = Pre Alarm
    - 2 = Main Alarm
- sensors
  - A string of sensors separated by a pipe (“|”)
- acousticAlarm
  - The state of the acoustic alarm 0 for not active and 1 for active
  - This only can be 1 if something is connected to the “Qh” Output in Loxone Config and the main alarm is active
- testAlarm
  - 0 if testalarm is not active and 1 if it is active
- alarmCause
  - Bitmask for Alarm-Causes (may be a combination)
    - 0x01: Smoke
    - 0x02: Water
    - 0x04: Temperature
    - 0x08: Arc Fault (electrical wiring)
- startTime
  - timestamp when alarm started

Commands

- mute
  - mutes the sirene
- quit
  - Acknowledge the alarm
- servicemode/{number}
  - number = Time in seconds until the service mode stops

SolarPumpController

Covered Config Items

- Solar Controller

Details

- buffers
  - List of used buffers
  - eg. [
    - {
      - “name”: “Buffer 1”
    - },

        - {
            - “name”: “Buffer2”
        - }
    -]

States

- bufferTemp{n}
    - n: number from 1 to 5
    - Temperature of buffer n
- bufferState{n}
    - n: number from 1 to 5
    - State of buffer n
    - Possible Values
        - 0 = Waiting, Buffer is waiting to be heated
        - 1 = Heating, Buffer is heating
        - 2 = Cooling, Buffer is cooling
        - 3 = OK, Buffer is heated to its temperature
- logicOverride
    - If the control is overwritten by logic
- collectorTemp
    - Temperature of the collector
- heatOverload
    - If heat overload is reached

SteakThermo

Covered Config Items

- Touch & Grill
    - Sensors are counted from left (yellow) to right (green)

Details

- deviceType
    - 0: No device connected
    - 1: Touch & Grill Air with two sensors
- isTouchProtectConnected
    - 0: No logic connected to the input DisT
    - 1: Logic connected to the input DisT
- isBrightnessConnected
    - 0: No logic connected to the parameter B
    - 1: Logic connected to the parameter B

States

- currentTemperatures
  - TextEvent can be interpreted as JSON
  - [
    - 82,
    - 148
  -]
  - Temperatures of the sensors, from left to right
- temperatureYellow
  - Temperature of the yellow sensor
- temperatureGreen
  - Temperature of the green sensor
- sensorInfo
  - TextEvent text can be interpreted as JSON
  - [
    - {
      - “name”: ”Brisket”,
      - “connected”: true,
      - “target”: 85
    - },
    - {
      - “name”: ”Meat”,
      - “connected”: true,
      - “target”: 100
    - }
  -]
  - name:
    - Name of the sensor, can be set by the user
  - connected
    - If the sensor is connected
  - target
    - defined target temperature of the sensor
  - Index in the Array
    - 0 = Left (yellow)
    - 1 = Right (green)
- targetYellow
  - Target temperature of the yellow sensor
- targetGreen
  - Target temperature of the green sensor
- sensorAlarms
  - TextEvent can be interpreted as JSON

    - [
        - {
            - “text”: “Target temperature reached”,
            - “time”: {secondsSince2009},
            - “ringing”: true
        - },
        - {
            - “text”: “Target temperature reached”,
            - “time”: {secondsSince2009},
            - “ringing”: false
        - }
    -]
    - text:
        - Description of the sensor alarm
    - time:
        - When the alarm has been triggered in seconds since 2009
    - ringing:
        - If the device is beeping or not
- yellowAlarmActive
    - If the yellow sensor does have an active alarm
- greenAlarmActive
    - If the green sensor does have an active alarm
- activeAlarmText
    - TextEvent the text property is the text of the currently active Alarm
    - Text is set if Qa is active and logic is connected Qa
    - If no alarm is active this value will be an empty string
- timerRemaining
    - Remaining time in seconds of the timer
- timerInfo
    - TextEvent, can be interpreted as JSON
    - {
        - “active”: true,
        - “duration”: 300
    - }
    - active:
        - If the timer is running
    - duration:
        - Time in Seconds
- timerAlarm
    - TextEvent, can be interpreted as JSON
    - {
        - “text”: “Timer started”,

        - “time”: {secondsSince2009},
        - “ringing”: true
    - }
    - text:
        - Description of the timer alarm
    - time:
        - When the alarm has been triggered in seconds since 2009
    - ringing:
        - If the device is beeping or not
- timerAlarmActive
    - If a timer alarm is active
- deviceState
    - The current state of the device
    - 0 = Running
    - 1 = Offline
    - 2 = Turned off
- displayAlwaysOnBat
    - If the display should stay on while the device is running on battery
- displayAlwaysOnDc
    - If the display should stay on while the device is connected to power
- availableControls
    - TextEvent, can be interpreted as JSON
    - [
        - {
            - “name”: “Kitchen”,
            - “uuid”: “0f38633e-0073-1a75-ffffed93b67b4c69”
        - },
        - {
            - “name”: “Garden”,
            - “uuid”: “0b734138-032f-0284-ffff403fb0c34b9e”
        - }
    -]
    - Defines the available controls the device has been assigned to
- activeControl
    - Index of the availableControls state
- isActive
    - If this particular control is the active control
- touchProtection
    - If the touch protection of the device is active
- displayBrightness
    - Brightness of the display, available since DEFINE_VERSION
    - Values from 0 - 100 (%)

-

Commands

- quitAlarm
  - Quits all ongoing alarms
- setSensor/{sensorIndex}/{targetTemp}/{sensorName}
  - sensorIndex
    - The index of the sensor
      - 0 = Left (yellow)
      - 1 = Right (green)
  - targetTemp
    - The target temperature of the sensor
  - sensorName
    - The name for the sensor
- setDisplayAlwaysOnBat/{on}
  - on
    - 0 = Disabled
    - 1 = Enabled
- setDisplayAlwaysOnDc/{on}
  - on
    - 0 = Disabled
    - 1 = Enabled
- setActive/{index}
  - Sets the given control as active
  - index
    - The index of the control in the availableControls state
- setThisActive
  - Sets this control as the active control
- setTimerDuration/{duration}
  - Sets the timer duration
  - duration
    - The duration in seconds
- startTimer
  - Starts the timer with the defined duration
- stopTimer
  - Stops the timer
- setTouchProtection/{on}
  - Enables or disables the touch protection
  - on
    - 0 = Disabled
    - 1 = Enabled
- setDisplayBrightness/{brightness}

- Sets the brightness of the display, available since DEFINE_VERSION
- brightness
  - Values from 0 to 100 (%)

Switch

Covered Config Items

- Virtual Input (Switch)
- Push button

States

- active
  - the current state of the switch

Commands

- on
  - activates the switch
- off
  - deactivates the switch

TextState

Covered Config Items

- State

States

- textAndIcon
  - TextEvent with text and icon

TimedSwitch

Covered Config Items

- Stairwell light switch
- Multifunction switch

Details

- isStairwayLs
    - true = “Stairwell light switch”
    - false = “Multifunction switch”

States

- deactivationDelayTotal
    - seconds, how long the output will be active if the timer is used.
- deactivationDelay
    - countdown until the output is deactivated.
        - 0 = the output is turned off
        - -1 = the output is permanently on
        - otherwise it will count down from deactivationDelayTotal

Commands

- on
    - Permanently activates the TimedSwitch, deactivationDelay will change to -1
- off
    - Turns off the TimedSwitch, deactivationDelay will change to 0
- pulse
    - deactivationDelay = 0
        - Will start the countdown, from deactivationDelayTotal to 0
    - isStairwayLs = true
        - deactivationDelay = -1
            - No effect, will remain permanently on.
        - deactivationDelay > 0
            - Restarts the countdown
    - isStairwayLs = false
        - turns it off. (from countdown or permanent on state)

Tracker

Covered Config Items

- Tracker

Details

- maxEntries

- maximal count of entries returned by the miniserver

States

- entries
  - entries returned from the miniserver as string
  - entries are separated by a pipe symbol “|”

UpDownLeftRight digital

Covered Config Items

- Virtual Input (Left-right buttons)
- Virtual Input (up-down buttons)

Commands

- UpOn
  - activates the up/right output
- UpOff
  - deactivates the up/right output
- PulseUp
  - since Config 8.0
  - impuls on up/right output
- DownOn
  - activates the down/left output
- DownOff
  - deactivates the down/left output
- PulseDown
  - since Config 8.0
  - impuls on down/left output

UpDownLeftRight analog

Covered Config Items

- Virtual Input (Left-Right buttons)
- Virtual Input (Up-Down buttons)

Details

- format
  - the format of the value
- min
  - the minimum value
- max
  - the maximum value
- step
  - the step to the next value of the virtual input when pressing up/down/left/right

States

- value
  - the current value of the virtual input
- error
  - indicates an invalid value of the virtual input

Commands

- {value}
  - value for the virtual input
  - between min and max

ValueSelector

Covered Config Items

- Push-button +/-
- Push-button +

Details

- increaseOnly
  - indicates if the button has only an “+”-input
- format
  - the format of the value

States

- min
  - the minimum value
- max
  - the maximum value
- step

    - the step to the next value of the virtual input when pressing up/down/left/right
- value
    - the current value of the virtual input

Commands

- {value}
    - value for the virtual input
    - between min and max

Ventilation

Covered Config Items

- Ventilation

Details

- hasPresence
    - indicates that the “Mv” input is connected to logic
- hasAbsenceMin
    - indicates that the parameter “V” is connected to logic
- hasAbsenceMax
    - indicates that the parameter “Vi” is connected to logic
- hasPresenceMin
    - indicates that the parameter “VP” is connected to logic
- hasPresenceMax
    - indicates that the parameter “VPi is connected to logic
- hasHumidityMax
    - indicates that the parameter “Hmax” is connected to logic
- hasIndorHumidity
    - indicates that the “Hi” input is connected to logic
- modes
    - All available modes
    - [
        - {
            - name: “Heat exchanger”
            - id: 2
        - },
        - {
            - name: “Exhaust”,
            - id: 1
        - },
        - ...

-]
- timerProfiles
  - Available timer profiles
  - [
    - {
      - name: “Resting”,
      - useCase: “”,
      - interval: 3600,
      - speed:
        - {
          - value: 100,
          - enabled: true
        - }
      - modes:
        - [
          - 0,
          - 1,
          - 4,
        -],
      - defaultMode: 1
    - },
    - ...
  -]
  - [useCase]
    - Optional description of the timer profile, string will be empty if no useCase is given
  - interval
    - Time in seconds this timer should be running
  - speed
    - Object with following properties
      - value
        - Speed in % of the ventilator
      - enabled
        - If the current timer profile is allowed to change the speed
  - modes
    - Array of available mode ids for this particular timer profile
    - These mode ids refer to the ids of the mode object in the modes detail
  - defaultMode
    - defines the per default selected mode for the timerProfile
  - In addition manual timers are available, the manual timer must be handled by the app itself, these are not provided by the Miniserver

- type
    - Defines the type of the control. This will be used to differentiate between different blocks to be able to use different UI or functions
        - 0: Generic
        - 1: Leaf

States

- ventReason
    - The ID for ventilation reason
        - 0: Basic ventilation
        - 1: Increased ventilation
        - 2: Reserved for future use
        - 3: Reserved for future use
        - 4: Stop
            - Block input “St”
        - 5: Window opened
            - Block input “Iw”
        - 6: Turbo
            - Block input “Tb”
        - 7: Manual
            - Overwritten via timer
        - 8: Exhaust
            - Block input “Ex”
        - 9: Fall-asleep-mode
            - Block input “Sl”
- temperatureSupport
    - Whether or not temperature support is active
    - -1: Cooling
    - 0: No Temperature support is active
    - 1: Heating
- activeTimerProfile
    - Index of the current active timer profile
    - -1: Manual
    - -2: No timer active
    - -3: Someone is changing settings
- stoppedBy
    - The name of the connected logic if the stop (“St”) input is active
- overwriteUntil
    - Unixtimestamp
    - This state is 0 if no timer is active
- controlInfo

        - TextEvent can be interpreted as JSON
        - {
            - level: 1,
            - title: “Error”,
            - desc: “Ventilation left offline”,
            - link: “https://www.loxone.com”,
            - action:
                - {
                    - name: “Acknowledge”,
                    - cmd: “cancelAlarm”
                - }
        - }
        - level
            - Defines the state of the control
            - 0: Ok
                - Won’t be displayed in the visualisation
            - 1: Error
                - Red colored
            - 2: Warning
                - Orange colored
            - 3: Info
                - Gray colored
        - [title]
            - Optional: The level will be used instead, if no title is defined
        - desc
            - Describes the current state
        - [link]
            - Optional: link to further information, e.g Online documentation
        - [action]
            - Optional: Defines an action object to resolve the current state
                - name: Name of the action
                - cmd: Command that resolves the current state
- speed
    - Value in % representing the speed of the ventilation
- mode
    - Defines the id of the current active mode defined in the details modes object
- presenceMin
    - Minimal procentual value of the ventilation if someone is present
- presenceMax
    - Maximal procentual value of the ventilation if someone is present
- absenceMin
    - Minimal procentual value of the ventilation if no one is present

- absenceMax
    - Maximal procentual value of the ventilation if no one is present
- humidityIndoor
    - Value of the indoor humidity sensor
- presence
    - If presence is active

Commands

- setTimer/{interval}/{speed}/{modeId}/{timerProfileIdx}
    - interval
        - Time in seconds
    - speed
        - Ventilation speed
        - 0 - 100
    - modeId
        - Id of the mode defined in the details modes object
    - timerProfileIdx
        - Array index of the timer
        - -1 for the Manual mode
- setAbsenceMin/{value}
    - Sets the minimal ventilation intensity if no one is present
- setAbsenceMax/{value}
    - Sets the maximal ventilation intensity if no one is present
- setPresenceMin/{value}
    - Sets the minimal ventilation intensity if someone is present
- setPresenceMax/{value}
    - Sets the maximal ventilation intensity if someone is present
- ackFilterChange
    - Acknowledges the “filter change” message

Webpage

Covered Config Items

- Webpage

Details

- url
    - The defined low resolution URL (This will be the Internal URL if you are connected internally and the external URL if you are connected externally)
- urlHd

- The defined high resolution URL (This will be the Internal URL if you are connected internally and the external URL if you are connected from externally)

WindowMonitor

Covered Config Items

Window- and Door Monitor

Details

- windows
  - An array of objects containing information on a window or a door monitored by this block.
  - The index of the object inside this array will correspond to the position of the state for this door or window in the windowStates array.
  - An object contains a name for the door or window and the uuid of the room it is in.

States

- windowStates
  - Will return a string containing the states of each of the windows and doors monitored by this block.
  - The individual states are separated by a comma. The position of the state in this string corresponds to the position of the object this state is for inside the windows-array in the details.
  - Each state is a integer value that represents a bitmask where the individual bits correspond to the following states:
    - none → state unknown / sensor offline
    - 1 → closed
    - 2 → tilted
    - 4 → open
    - 8 → locked
    - 16 → unlocked
- numOpen, numClosed, numTilted, numOffline, numLocked, numUnlocked
  - The number of windows & doors in the corresponding states.
  - The sum of the values from all these states is equal to the number of windows & doors monitored.
  - The windows/doors with two states will always be counted to the “worst” state.

- e.g.: A lockable door is unlocked and closed. It will be counted to numUnlocked and not to numClosed.