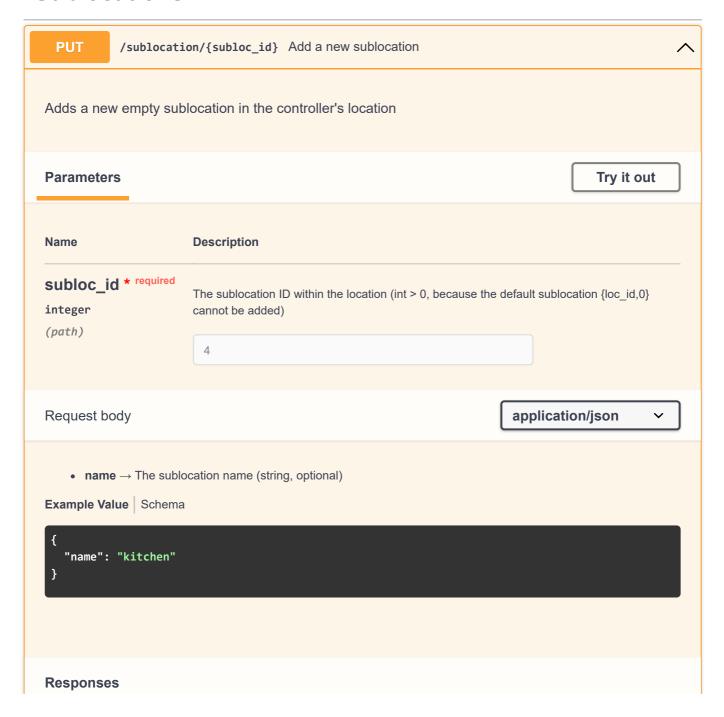
JANET Controller REST API OASS OASS

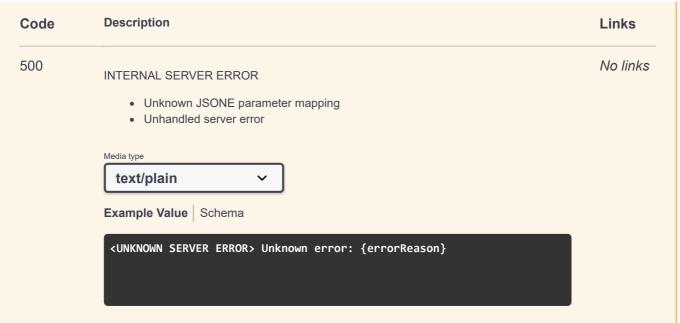
REST API for interfacing with a JANET Controller node in the JANET Home Simulation System

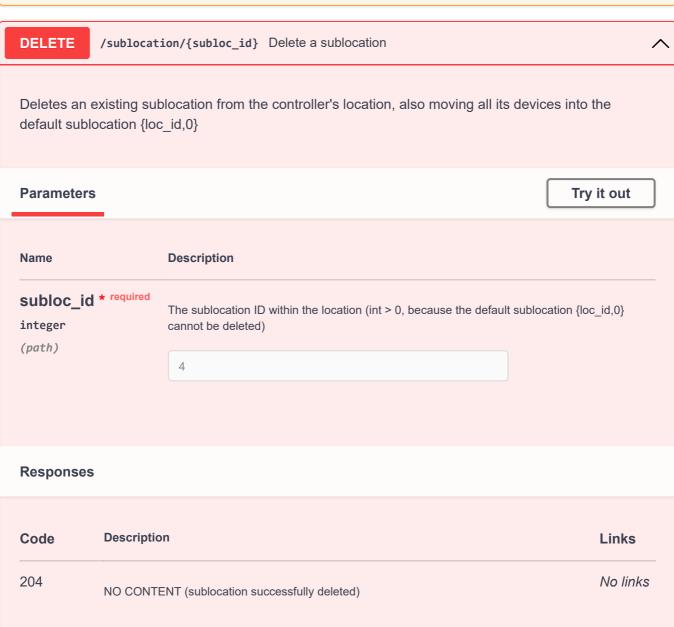


Sublocations API



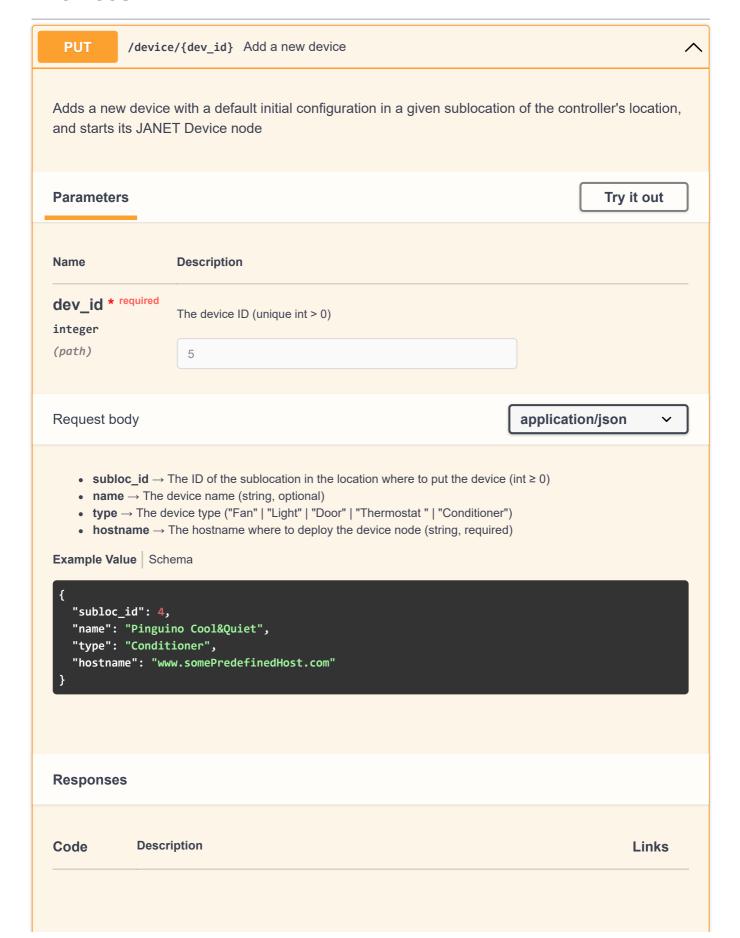




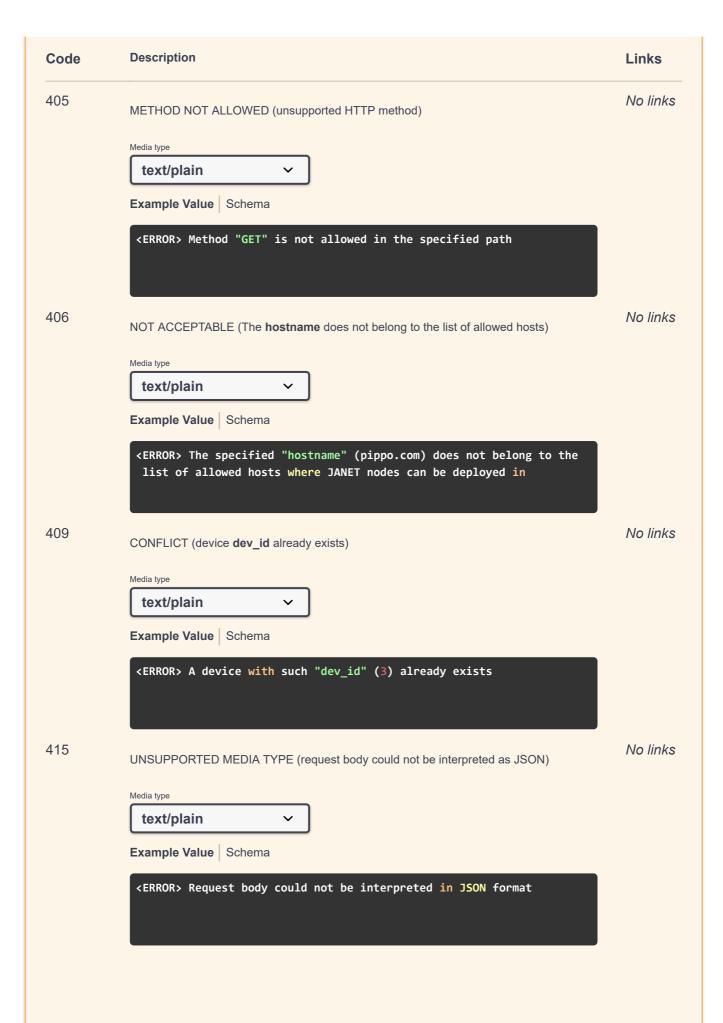


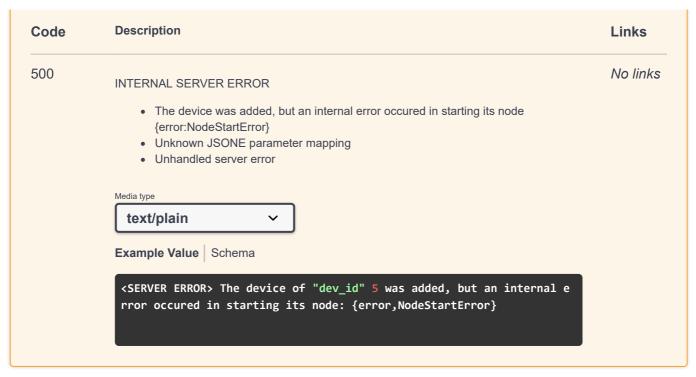


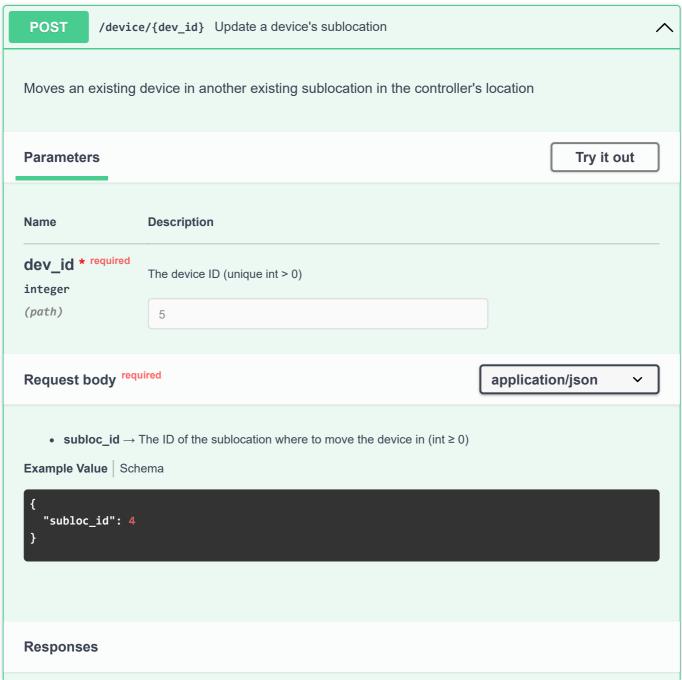
Devices API



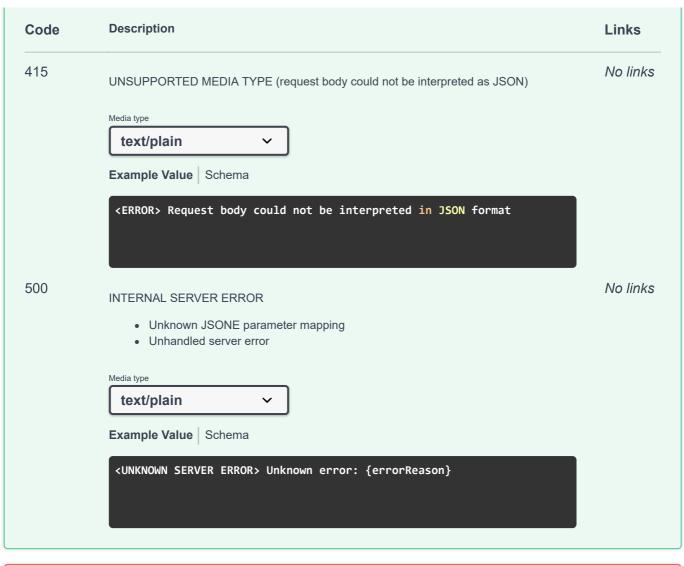


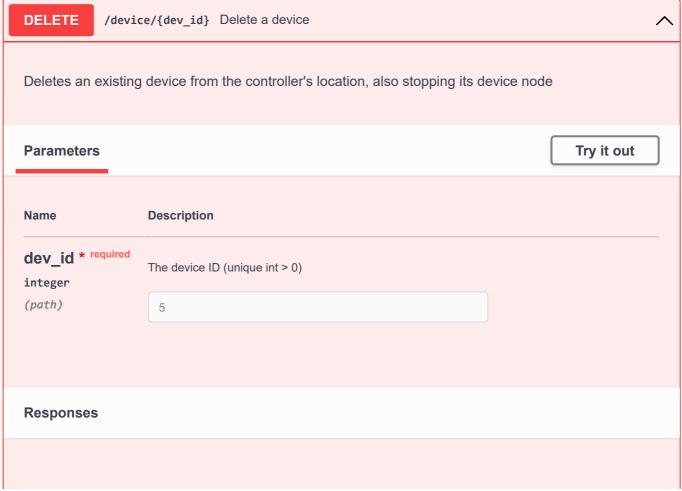


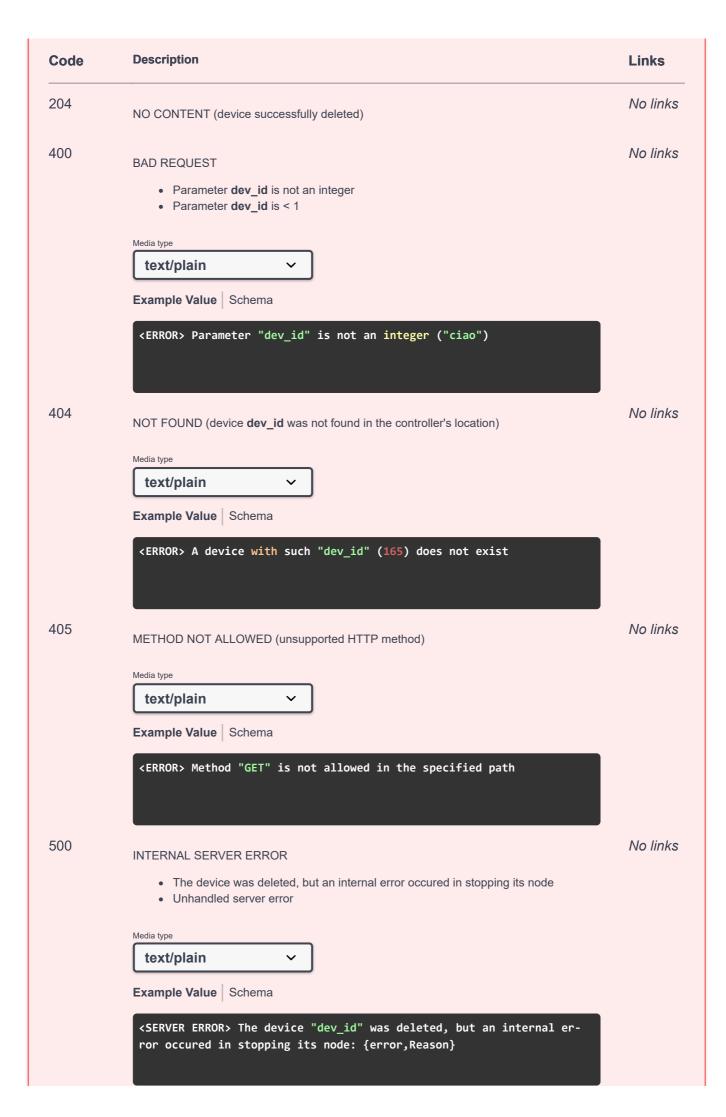












Devices Commands API

PATCH

/devcommands Issue device commands

Issue one or more commands to the devices in the controller's location

Request Body

The request body must consist in a list of "N" **Device Commands (DevCommands)** structured as follows:

```
// Device Command 1
  "dev_id": 5, // Device the command refers to
  "actions":
              // Set of device-specific actions
              // (in this example, for a fan)
    "onOff":"on",
   "fanSpeed":50
 },
 // Device Command 2
  "dev_id": 6,
  "actions": // Note that not every device-specific action
              // is required in the "actions" object
   "tempTarget":25
 },
// Device Command N
{ ... }
```

Where only one command towards the same "dev_id" can be issued, and the list of device-specific actions with their allowed values can be found in the *actions*.* schemas.

HTTP Response Format

Response Body

The response body consists in a list of "N" **Device Responses** (**DevResponses**), each associated with a device command (but not necessarily in the same order) and that can be **Successful** or **Unsuccessful**:

Successful Device Response

Where the "updatedState" object may consist in:

- If the command was issued within a predefined maximum time from the last update the controller received from the device, in the traits that have changed from such update (possibly *none*, leading to an empty object "updatedState":{})
- If the command was issued after a predefined maximum time from the last update the controller received from the device, in the complete device state (which in this case cannot be empty)

Unsuccessful Device Response

Where the schema of the "state" object is analogous to the one of the "actions" objects for the device commands, and the full list of errors that may be returned can be found in the *errorReason.** schemas.

Response Status Code

Apart from corner cases where the request could not be interpreted correctly, the HTTP status code of the response is set to:

- 200 (OK) -> ALL device commands were successful (all device responses are successful)
- 202 (ACCEPTED) -> There are both successful and unsuccessful commands
- maxErr -> The maximum status code among all unsuccessful device responses (if no command was successful)

Parameters

Try it out

No parameters

Request body required

application/json

Example Value | Schema

Responses

Code Description Links

200 OK (ALL device commands were successful) No links

ON (ALL device commands were successful)

Media type

application/json ~

Controls Accept header.

Example Value | Schema

```
[
    "dev_id": 5,
    "status": 200,
    "devstate": {
        "onOff": "off",
        "fanSpeed": 50
     },
    "timestamp": "2021-10-06T02:49:35"
}
]
```

Code **Description** Links 202 No links ACCEPTED (There are both successful and unsuccessful responses) Media type application/json Example Value | Schema "dev_id": 5, "status": 200, "devstate": { "onOff": "off", "fanSpeed": 50 "timestamp": "2021-10-06T02:49:35" }, "dev_id": 5, "status": 400, "errorReason": "A valid command towards such device was already issued" No links 400 **BAD REQUEST** • The request body could not be interpreted as a JSON list • The request body is empty Media type text/plain Example Value | Schema <ERROR> The request body could not be interpreted as a JSON list 405 No links METHOD NOT ALLOWED (unsupported HTTP method) Media type text/plain Example Value | Schema <ERROR> Method "GET" is not allowed in the specified path



```
Code
             Description
                                                                                                Links
4XX
                                                                                                No links
             (all commands were unsuccessful, the one with the highest code is "4XX")
             Media type
               application/json
             Example Value | Schema
                   "dev_id": 5,
                   "status": 400,
                   "errorReason": "A valid command towards such device was already
               issued"
                                                                                                No links
5XX
             (all commands were unsuccessful, the one with the highest code is "5XX")
             Media type
               application/json
             Example Value | Schema
                   "dev_id": 5,
                   "status": 400,
                   "errorReason": "A valid command towards such device was already
               issued"
```

```
dev_id integer
minimum: 1
example: 5
```

```
actions < {
   oneOf ->
                         actions.fan > {...}
                         minProperties: 1
                         maxProperties: 2
                         actions.light > {...}
                         minProperties: 1
                         maxProperties: 3
                         actions.door > {...}
                         minProperties: 1
                         maxProperties: 2
                         actions.thermostat > {...}
                         minProperties: 1
                         maxProperties: 2
                         actions.conditioner > {...}
                         minProperties: 1
                         maxProperties: 3
example: OrderedMap { "onOff": "on", "fanSpeed": 30 }
```

```
actions.light < {
   anyOf ->
                         trait.onOff string
                          example: off
                          Enum:

∨ [ on, off ]
                          trait.brightness integer
                          minimum: 1
                          maximum: 100
                          example: 50
                          trait.color string
                          example: #FFFFF
minProperties: 1
maxProperties: 3
actions.door < {
   anyOf ->
                          trait.openClose string
                          example: close
                          Enum:

∨ [ open, close ]
                          trait.lockUnlock string
                          example: unlock
                          Enum:
                           ▼ [ lock, unlock ]
minProperties: 1
maxProperties: 2
actions.thermostat ∨ {
   anyOf ->
                          trait.onOff string
                          example: off
                          Enum:

∨ [ on, off ]
                          trait.tempTarget integer
                          minimum: 0
                          maximum: 50
                          example: 21
```

minProperties: 1
maxProperties: 2

```
actions.conditioner < {
   anyOf ->
                         trait.onOff string
                         example: off
                         Enum:

∨ [ on, off ]
                         trait.tempTarget integer
                         minimum: 0
                         maximum: 50
                         example: 21
                         trait.fanSpeed integer
                         minimum: 1
                         maximum: 100
                         example: 50
}
minProperties: 1
maxProperties: 3
action.onOff string
example: off
Enum:
 ▼ [ on, off ]
action.fanSpeed integer
minimum: 1
maximum: 100
example: 50
action.brightness integer
minimum: 1
maximum: 100
example: 50
action.color string
example: #FFFFF
action.openClose string
example: close
Enum:

∨ [ open, close ]
```

```
action.lockUnlock string
example: unlock
Enum:
 ▼ [ lock, unlock ]
action.tempTarget integer
minimum: 0
maximum: 50
example: 21
devresponse_successful > {
   dev_id
                         dev_id integer
                         minimum: 1
                         example: 5
   status
                         integer
                         minimum: 200
                         maximum: 200
                         example: 200
   devstate
                         devstate > {...}
   timestamp
                         string
                         example: 2021-10-06T02:49:35
}
devresponse_unsuccessful > {
   dev_id
                         dev_id integer
                         minimum: 1
                         example: 5
                         integer
   status
                         minimum: 307
                         maximum: 504
                         example: 400
   errorReason
                         errorReason > {...}
                         example: A valid command towards such device was already issued
```

}

```
devstate.door > {
   openClose*
                         trait.openClose string
                         example: close
                         Enum:

✓ [ open, close ]
   lockUnlock*
                         trait.lockUnlock string
                         example: unlock
                         Enum:

✓ [ lock, unlock ]
}
devstate.thermostat ∨ {
   onOff*
                         trait.onOff string
                         example: off
                         Enum:
                           ▼ [ on, off ]
   tempTarget*
                         trait.tempTarget integer
                         minimum: 0
                         maximum: 50
                         example: 21
   tempCurrent*
                         trait.tempCurrent integer
```

```
\leftarrow
devstate.conditioner > {
   onOff*
                          trait.onOff string
                          example: off
                          Enum:

∨ [ on, off ]
   tempTarget*
                          trait.tempTarget integer
                          minimum: 0
                          maximum: 50
                          example: 21
   tempCurrent*
                          trait.tempCurrent integer
                          example: 21
   fanSpeed*
                          trait.fanSpeed integer
                          minimum: 1
                          maximum: 100
                          example: 50
}
```

example: 21

}

```
errorReason.notaninteger_devid string
Enum:

∨ [ Parameter 'dev_id' is not an integer ]
errorReason.outofrange_devid string
Enum:
 errorReason.duplicate_dev_id string
Enum:

▼ [ A valid command towards such device was already issued ]

errorReason.missingparam_actions string
Enum:

∨ [ The 'actions' object is missing ]
errorReason.notamap_actions string
Enum:

▼ [ The 'actions' parameter could not be interpreted as a
JSON object ]
errorReason.empty_actions string
Enum:

∨ [ The 'actions' object is empty ]
errorReason.device_not_exists string
Enum:

▼ [ A device with such 'dev_id' does not exist ]
errorReason.device_offline string
Enum:

▼ [ The device is currently offline ]
errorReason.invalid_action string
Enum:

▼ [ The 'actions' object contains invalid actions for the
associated device type (fan) ]
errorReason.invalid_state_door string
Enum:

▼ [ The state {'open', 'lock'} is invalid for device type
'door' ]
errorReason.unknown_jsone_cast string
Enum:

▼ [ Invalid value of trait 'fanSpeed' was cast to an unknown
term() by the JSONE library ]
errorReason.invalid_db_devtype string
Enum:

▼ [ The device type in the controller database (fAn) is
invalid ]
errorReason.unhandled_parsing string
Enum:

▼ [ Unhandler server error while parsing the command:
{error,Reason} ]
errorReason.cmdclient_crash string
Enum:

▼ [ Server error in issuing the device commands:
{cmdclient_crash} ]
errorReason.unhandled_issuing_general string

▼ [ Server error in issuing the device commands:
{error,Reason} ]
errorReason.invalid_devconfig string
Enum:
```

```
oldsymbol{\lor} [ The command was rejected by the device for it is invalid
                          in its current state ]
                          errorReason.dev_timeout string
                          Enum:

▼ [ The device node is not responding ]
                          errorReason.statem_timeout string

▼ [ The device state machine is not responding ]
                          errorReason.unexpected_response string

▼ [ An unexpected device command response was received:
                          {UnexpectedResponse} ]
                          errorReason.missing_response string
                          Enum:

▼ [ No command response was returned from the device ]
                          errorReason.invalid_devtype string
                          Enum:

▼ [ The updated state returned by the device is malformed ]

                          errorReason.unhandled_issuing string
                          Enum:

▼ [ Unhandled server error in parsing the command response:
                          {error,Reason} ]
example: A valid command towards such device was already issued
errorReason.missingparam_devid string

∨ [ Required parameter 'dev_id' is missing ]
errorReason.notaninteger_devid string
Enum:

∨ [ Parameter 'dev_id' is not an integer ]
errorReason.outofrange_devid string
Enum:

∨ [ Invalid value of parameter 'dev_id' (0) ]
errorReason.duplicate_dev_id string

▼ [ A valid command towards such device was already issued ]
```

```
errorReason.missingparam_actions string
Enum:

∨ [ The 'actions' object is missing ]
errorReason.notamap_actions string
Enum:

▼ [ The 'actions' parameter could not be interpreted as a JSON object ]

errorReason.empty_actions string
Enum:

∨ [ The 'actions' object is empty ]
errorReason.device_not_exists string

∨ [ A device with such 'dev_id' does not exist ]
errorReason.device_offline string

▼ [ The device is currently offline ]
                                                                                    \leftarrow
errorReason.invalid action string

▼ [ The 'actions' object contains invalid actions for the associated device type (fan) ]

errorReason.invalid_state_door string
Enum:

▼ [ The state {'open', 'lock'} is invalid for device type 'door' ]

errorReason.unknown_jsone_cast string

▼ [ Invalid value of trait 'fanSpeed' was cast to an unknown term() by the JSONE library
```

```
errorReason.invalid_db_devtype string
Enum:

▼ [ The device type in the controller database (fAn) is invalid ]

errorReason.unhandled_parsing string
Enum:
 ▼ [ Unhandler server error while parsing the command: {error,Reason} ]
errorReason.cmdclient_crash string
 ▼ [ Server error in issuing the device commands: {cmdclient_crash} ]
errorReason.unhandled_issuing_general string

▼ [ Server error in issuing the device commands: {error,Reason} ]

errorReason.invalid_devconfig string
 ullet [ The command was rejected by the device for it is invalid in its current state ]
errorReason.dev_timeout string

▼ [ The device node is not responding ]
errorReason.statem timeout string
Enum:

▼ [ The device state machine is not responding ]
errorReason.unexpected_response string
Enum:

▼ [ An unexpected device command response was received: {UnexpectedResponse} ]
```

```
errorReason.missing_response string
Enum:

▼ [ No command response was returned from the device ]
errorReason.invalid_devtype string
Enum:
 ullet [ The updated state returned by the device is malformed ]
errorReason.unhandled_issuing string
 ▼ [ Unhandled server error in parsing the command response: {error,Reason} ]
trait.onOff string
example: off
Enum:

∨ [ on, off ]
trait.fanSpeed integer
minimum: 1
maximum: 100
example: 50
                                                                                     \leftarrow
trait.brightness integer
minimum: 1
maximum: 100
example: 50
trait.color string
example: #FFFFF
trait.openClose string
example: close
Enum:

∨ [ open, close ]
```

