Smart Home

Overview

This is the documentation for smart home web rest api by reem alghamdi

Version information

Version : 1.0

Contact information

Contact Email : reem.brain@gmail.com

License information

License : MIT
License URL : https://opensource.org/licenses/MIT
Terms of service : null

URI scheme

Host : gp.reem-codes.com
BasePath : /api
Schemes : HTTPS, HTTP

Tags

- Command : What user wants rapberry to do. It is a mapping between the time, the hardware and the configuration
- Configuration : All the possible configuration and status for a given hardware. eg: on, off, red, extended ...
- Hardware : Everything about the sensors and actuators connected to raspberry pi
- Response : What the raspberry wants the user to know. It is a mapping between the time of execution, the command triggered and whether the action was done or not
- Schedule : scheduling information for command if any. Specifys the days and time of day the user would like a command to be triggered

Paths

Add a new command

```
POST /command
```

Parameters

Type	Name	Description	Schema
Body	**command** *optional*	The command to create.	command

command

Name	Description	Schema
configurationId *required*	the id of the configuration to apply to the hardware **Example** : 3	integer
hardwareId *required*	the id of the hardware this command targets **Example** : 1	integer
scheduleId *optional*	the id of the schedule if any **Example** : 453	integer

Responses

HTTP Code	Description	Schema
201	Created	Command
405	Invalid input	No Content

Consumes

- application/json

Produces

- application/json

Tags

- Command

get all commands

```
GET /command
```

Responses

HTTP Code	Description	Schema
200	return an array of command objects	< Command > array

Produces

- `application/json`

Tags

- Command

get a command by id

```
GET /command/{commandId}
```

Parameters

Type	Name	Schema
Path	**commandId** *required*	integer

Responses

HTTP Code	Description	Schema
200	get the command	Command
404	Not found	No Content

Produces

- `application/json`

Tags

- Command

edit an existing command given its id

```
PUT /command/{commandId}
```

Parameters

Type	Name	Description	Schema
Path	**commandId** *required*		integer
Body	**command** *optional*	The command to edit.	command

command

Name	Description	Schema
configurationId *required*	the id of the configuration to apply to the hardware **Example** : 3	integer
hardwareId *required*	the id of the hardware this command targets **Example** : 1	integer
scheduleId *optional*	the id of the schedule if any **Example** : 453	integer

Responses

HTTP Code	Description	Schema
200	Edited	Command
404	Not found	No Content
405	Invalid input	No Content

Consumes

- `application/json`

Produces

- `application/json`

Tags

- Command

delete a command by id

```
DELETE /command/{commandId}
```

Parameters

Type	Name	Schema
Path	**commandId** *required*	integer

Responses

HTTP Code	Description	Schema
204	deleted	No Content
404	Not found	No Content

Tags

- Command

Add a new configuration to a hardware

```
POST /configuration
```

Parameters

Type	Name	Description	Schema
Body	**configuration** *optional*	The configuration to create.	configuration

configuration

Name	Description	Schema
description *optional*	information about the configuration **Example** : `"EXTEND means that it will become longer by 3cm"`	string
hardwareId *required*	the hardware this configuration belongs to **Example** : `"ON belongs to the hardwareID 1, which is RED LED"`	integer
name *required*	the configuration name **Example** : `"ON"`	string

Responses

HTTP Code	Description	Schema
201	Created	Configuration
405	Invalid input	No Content

Consumes

- `application/json`

Produces

- `application/json`

Tags

- Configuration

get a configuration by id

```
GET /configuration/{configurationId}
```

Parameters

Type	Name	Schema
Path	**configurationId** *required*	integer

Responses

HTTP Code	Description	Schema
200	get the configuration	Configuration
404	Not found	No Content

Produces

- `application/json`

Tags

- Configuration

edit an existing configuration given its id

```
PUT /configuration/{configurationId}
```

Parameters

Type	Name	Description	Schema
Path	**configurationId** *required*		integer
Body	**configuration** *optional*	The configuration to edit.	configuration

configuration

Name	Description	Schema
description *optional*	information about the configuration **Example** : `"EXTEND means that it will become longer by 3cm"`	string

Name	Description	Schema
hardwareId *optional*	the hardware this configuration belongs to **Example** : `"ON belongs to the hardwareID 1, which is RED LED"`	integer
name *optional*	the configuration name **Example** : `"ON"`	string

Responses

HTTP Code	Description	Schema
200	Edited	Configuration
404	Not found	No Content
405	Invalid input	No Content

Consumes

- `application/json`

Produces

- `application/json`

Tags

- Configuration

delete a configuration by id

```
DELETE /configuration/{configurationId}
```

Parameters

Type	Name	Schema
Path	**configurationId** *required*	integer

Responses

HTTP Code	Description	Schema
204	deleted	No Content
404	Not found	No Content

Tags

- Configuration

Add a new Hardware to the system

```
POST /hardware
```

Parameters

Type	Name	Description	Schema
Body	**hardware** *optional*	The hardware to create.	hardware

hardware

Name	Description	Schema
description *optional*	additional info regarding the hardware **Example** : `"LED is a small electrical component that can emit light"`	string
gpio *required*	the gpio pin the hardware is installed at **Example** : `11`	integer
icon *optional*	URL image to desired icon in client **Example** : `"https://image.flaticon.com/icons/png/512/32/32750.png"`	string
name *required*	the hardware name **Example** : `"RGB LED"`	string

Responses

HTTP Code	Description	Schema
201	Created	Hardware
405	Invalid input	No Content

Consumes

- `application/json`

Produces

- `application/json`

Tags

- Hardware

get all hardwares connected to raspberry pi

```
GET /hardware
```

Responses

HTTP Code	Description	Schema
200	return an array of hardware objects	< Hardware > array

Produces

- `application/json`

Tags

- Hardware

get a hardware by id

```
GET /hardware/{hardwareId}
```

Parameters

Type	Name	Schema
Path	**hardwareId** *required*	integer

Responses

HTTP Code	Description	Schema
200	get the hardware	Hardware
404	Not found	No Content

Produces

- `application/json`

Tags

- Hardware

edit an existing hardware given its id

```
PUT /hardware/{hardwareId}
```

Parameters

Type	Name	Description	Schema
Path	**hardwareId** *required*		integer
Body	**hardware** *optional*	The hardware to edit.	hardware

hardware

Name	Description	Schema
description *optional*	additional info regarding the hardware **Example** : `"LED is a small electrical component that can emit light"`	string

Name	Description	Schema
gpio *required*	the gpio pin the hardware is installed at **Example** : `11`	integer
icon *optional*	URL image to desired icon in client **Example** : `"https://image.flaticon.com/icons/png/512/32/32750.png"`	string
name *required*	the hardware name **Example** : `"RGB LED"`	string

Responses

HTTP Code	Description	Schema
200	Edited	Hardware
404	Not found	No Content
405	Invalid input	No Content

Consumes

- `application/json`

Produces

- `application/json`

Tags

- Hardware

delete a hardware by id

```
DELETE /hardware/{hardwareId}
```

Parameters

Type	Name	Schema
Path	**hardwareId** *required*	integer

Responses

HTTP Code	Description	Schema
204	deleted	No Content
404	Not found	No Content

Tags

- Hardware

get all commands given a hardwareID

```
GET /hardware/{hardwareId}/command
```

Parameters

Type	Name	Schema
Path	**hardwareId** *required*	integer

Responses

HTTP Code	Description	Schema
200	return an array of command objects	< Command > array

Produces

- `application/json`

Tags

- Command
- Hardware

get all configuration for a given hardware

```
GET /hardware/{hardwareId}/configuration
```

Parameters

Type	Name	Schema
Path	**hardwareId** *required*	integer

Responses

HTTP Code	Description	Schema
200	return an array of configuration objects	< Configuration > array

Produces

- `application/json`

Tags

- Configuration
- Hardware

Add a new response

```
POST /response
```

Parameters

Type	Name	Description	Schema
Body	**response** *optional*	The response to create.	response

response

Name	Description	Schema
commandId *required*	the command id the response is for **Example** : 23	integer
executionTime *optional*	the actual time raspberry pi executed the command	string (date-time)

Name	Description	Schema
isDone *required*	whether the command has been successfully done or not **Example** : `true`	boolean
message *optional*	optional message regarding the action **Example** : `"the command 243 was successfully executed!"`	string

Responses

HTTP Code	Description	Schema
201	Created	Response
405	Invalid input	No Content

Consumes

- `application/json`

Produces

- `application/json`

Tags

- Response

get all responses

```
GET /response
```

Responses

HTTP Code	Description	Schema
200	return an array of response objects	< Response > array

Produces

- `application/json`

Tags

- Response

get a response by id

```
GET /response/{responseId}
```

Parameters

Type	Name	Schema
Path	**responseId** *required*	integer

Responses

HTTP Code	Description	Schema
200	get the response	Response
404	Not found	No Content

Produces

- `application/json`

Tags

- Response

edit an existing response given its id

```
PUT /response/{responseId}
```

Parameters

Type	Name	Description	Schema
Path	**responseId** *required*		integer

Type	Name	Description	Schema
Body	**response** *optional*	The response to edit.	response

response

Name	Description	Schema
commandId *optional*	the command id the response is for **Example** : `23`	integer
executionTime *optional*	the actual time raspberry pi executed the command	string (date-time)
isDone *optional*	whether the command has been successfully done or not **Example** : `false`	boolean
message *optional*	optional message regarding the action **Example** : `"the command 243 was successfully executed!"`	string

Responses

HTTP Code	Description	Schema
200	Edited	Response
404	Not found	No Content
405	Invalid input	No Content

Consumes

- `application/json`

Produces

- `application/json`

Tags

- Response

delete a response by id

```
DELETE /response/{responseId}
```

Parameters

Type	Name	Schema
Path	**responseId** *required*	integer

Responses

HTTP Code	Description	Schema
204	deleted	No Content
404	Not found	No Content

Tags

- Response

Add a new schedule

```
POST /schedule
```

Parameters

Type	Name	Description	Schema
Body	**schedule** *optional*	The schedule to create.	schedule

schedule

Name	Description	Schema
commandId *required*	the command id this schedule belongs to **Example** : 231	integer

Name	Description	Schema
days *optional*	value between 0 and 127, representing 7 bits each bit correspond to a day in the order: sun mon tues web thurs fri sat **Minimum value** : `0` **Maximum value** : `127` **Example** : `120`	integer (int64)
time *optional*	the time of day the command shall be executed **Example** : `"13:50"`	string (time)

Responses

HTTP Code	Description	Schema
201	Created	Schedule
405	Invalid input	No Content

Consumes

- `application/json`

Produces

- `application/json`

Tags

- Schedule

get all schedules

```
GET /schedule
```

Responses

HTTP Code	Description	Schema
200	return an array of schedule objects	< Schedule > array

Produces

- application/json

Tags

- Schedule

get a schedule by id

```
GET /schedule/{scheduleId}
```

Parameters

Type	Name	Schema
Path	**scheduleId** *required*	integer

Responses

HTTP Code	Description	Schema
200	get the schedule	Schedule
404	Not found	No Content

Produces

- application/json

Tags

- Schedule

edit an existing schedule given its id

```
PUT /schedule/{scheduleId}
```

Parameters

Type	Name	Description	Schema
Path	**scheduleId** *required*		integer
Body	**schedule** *optional*	The schedule to edit.	schedule

schedule

Name	Description	Schema
commandId *required*	the command id this schedule belongs to **Example** : `231`	integer
days *optional*	value between 0 and 127, representing 7 bits each bit correspond to a day in the order: sun mon tues web thurs fri sat **Minimum value** : `0` **Maximum value** : `127` **Example** : `120`	integer (int64)
time *optional*	the time of day the command shall be executed **Example** : `"13:50"`	string (time)

Responses

HTTP Code	Description	Schema
200	Edited	Schedule
404	Not found	No Content
405	Invalid input	No Content

Consumes

- `application/json`

Produces

- `application/json`

Tags

- Schedule

delete a schedule by id

```
DELETE /schedule/{scheduleId}
```

Parameters

Type	Name	Schema
Path	**scheduleId** *required*	integer

Responses

HTTP Code	Description	Schema
204	deleted	No Content
404	Not found	No Content

Tags

- Schedule

Definitions

Command

Name	Description	Schema
configuration *required*	the configuration desired for the hardware	Configuration
hardware *required*	the hardware this command is issued for	Hardware
id *required*	the identifier for the command in the database **Example** : `456`	integer (int64)
schedule *optional*	the schedule for this command. If none this command is immediate	Schedule

Name	Description	Schema
updateAt *optional*	time of creation/last updating	string (date-time)

Configuration

Name	Description	Schema
description *optional*	information about the configuration **Example** : `"EXTEND means that it will become longer by 3cm"`	string
hardware *required*	the hardware this configuration belongs to	Hardware
id *required*	the identifier for the configuration in the database **Example** : `3`	integer (int64)
name *required*	name of the configuration **Example** : `"ON"`	string
updateAt *optional*	time of creation/last updating	string (date-time)

Hardware

Name	Description	Schema
description *optional*	additional info regarding the hardware **Example** : `"LED is a small electrical component that can emit light"`	string
gpio *required*	the gpio pin the hardware is installed at **Example** : `11`	integer
icon *optional*	URL image to desired icon in client **Example** : `"https://image.flaticon.com/icons/png/512/32/32750.png"`	string
id *required*	the identifier for the configuration in the database **Example** : `1`	integer (int64)
name *required*	the hardware name **Example** : `"RGB LED"`	string

Name	Description	Schema
status *optional*	the current configuration for the hardware	Configuration
updateAt *optional*	time of creation/last updating	string (date-time)

Response

Name	Description	Schema
command *required*	the command executed resulting in this response	Command
executionTime *optional*	the actual time raspberry pi executed the command	string (date-time)
id *required*	the identifier for the response in the database **Example** : `865`	integer (int64)
isDone *required*	whether the command has been successfully done or not **Example** : `false`	boolean
message *optional*	optional message regarding the action **Example** : `"electrical error: hardware is not connected to the circuit"`	string
updateAt *optional*	time of creation/last updating	string (date-time)

Schedule

Name	Description	Schema
command *required*	the command where this scheduling info is for	Command
days *optional*	value between 0 and 127, representing 7 bits each bit correspond to a day in the order: sun mon tues web thurs fri sat **Minimum value** : `0` **Maximum value** : `127` **Example** : `120`	integer (int64)

Name	Description	Schema
id *required*	the identifier for the schedule in the database **Example** : `75`	integer (int64)
time *optional*	the time of day the command shall be executed **Example** : `"13:50"`	string (time)
updateAt *optional*	time of creation/last updating	string (date-time)