



Version: 1.0

# Temperature

The Temperature component handles the monitoring of device's temperature sensors.

Temperature components are identified with `temperature:<id>` in objects containing multiple component payloads.

The Temperature component uses `Temperature` as RPC namespace and implements the minimal component interface:

- `Temperature.GetConfig` to obtain the component's [configuration](#)
- `Temperature.SetConfig` to update the component's [configuration](#)
- `Temperature.GetStatus` to obtain the component's [status](#)

## Methods

### Temperature.SetConfig

Properties:

Property	Type	Description
<code>id</code>	<i>number</i>	Id of the Temperature component instance
<code>config</code>	<i>object</i>	Configuration that the method takes

Find more about the config properties in [config section](#)

### Temperature.GetConfig

Properties:

Property	Type	Description
<code>id</code>	<i>number</i>	Id of the Temperature component instance

Find the `Temperature.GetConfig` response properties in [config section](#)

## Temperature.GetStatus

Properties:

Property	Type	Description
<code>id</code>	<i>number</i>	Id of the Temperature component instance

Find more about the status response properties in [status section](#)

## Configuration

The configuration of the Temperature component allows to adjust the temperature report threshold value. To Get/Set the configuration of the Temperature component its `id` must be specified.

Properties:

Property	Type	Description
<code>id</code>	<i>number</i>	Id of the Temperature component instance
<code>name</code>	<i>string or null</i>	Name of the Temperature instance. <code>name</code> length should not exceed 64 chars
<code>report_thr_C</code>	<i>number</i>	Temperature report threshold in Celsius. Accepted range is device-specific, default [0.5..5.0]C unless specified otherwise

Property	Type	Description
<code>offset_C</code>	<i>number</i>	Offset in Celsius to be applied to the measured temperature. Accepted range is device-specific, default [-50.0 .. 50.0] unless specified otherwise

## Status

The status of the Temperature component represents the measurement of the associated temperature sensor. To obtain the status of the Temperature component its `id` must be specified.

Property	Type	Description
<code>id</code>	<i>number</i>	Id of the Temperature component instance
<code>tC</code>	<i>number or null</i>	Temperature in Celsius ( <code>null</code> if valid value could not be obtained)
<code>tF</code>	<i>number or null</i>	Temperature in Fahrenheit ( <code>null</code> if valid value could not be obtained)
<code>errors</code>	<i>array of type string</i>	Shown only if at least one error is present. May contain <code>out_of_range</code> , <code>read</code> when there is problem reading sensor

## Webhook Events

There are two events related to the Temperature component that can trigger webhooks:

- `temperature.change` - produced when temperature delta between two measurements is greater than `report_thr_C`
- `temperature.measurement` - produced on a monotonic measurement period (60s)

`temperature.change` and `temperature.measurement` support two *attributes*, that can be used to compose conditional [webhooks](#):

Property	Type	Description
<code>tC</code>	<i>number</i>	New temperature in Celsius
<code>tF</code>	<i>number</i>	New temperature in Fahrenheit

## Examples

### Temperature.SetConfig example

**Temperature.SetConfig**  
**HTTP GET Request**

**Temperature.SetConfig**  
**Curl Request**

**Temperature.SetConfig**  
**Mos Request**

```
http://192.168.33.1/rpc/Temperature.SetConfig?id=0&config=
{"name": "Temperature0"}
```

#### Response

```
{
  "restart_required": false
}
```

### Temperature.GetConfig example

**Temperature.GetConfig**  
**HTTP GET Request**

**Temperature.GetConfig**  
**Curl Request**

**Temperature.GetConfig**  
**Mos Request**

```
http://192.168.33.1/rpc/Temperature.GetConfig?id=0
```

#### Response

```
{
  "id": 0,
  "name": null,
  "report_thr_C": 0.5,
  "offset_C": 0
}
```

## Temperature.GetStatus example

**Temperature.GetStatus**  
**HTTP GET Request**

**Temperature.GetStatus**  
**Curl Request**

**Temperature.GetStatus**  
**Mos Request**

```
http://192.168.33.1/rpc/Temperature.GetStatus?id=0
```

### Response

```
{
  "id": 0,
  "tC": 24.4,
  "tF": 75.9
}
```