EiT API

Device status

Resource for storing and fetching the status of a device with a given id.

Get device status

0.1 GET /status/{device}

Request *raw*

Response

```
200 (OK)
Content-Type: application/json
```

```
    {
        "timestamp": "Timestamp in milliseconds when the server received the last status
update",
        "_id": "Database id, not needed for anything",
        "device_id": "The same as the {device}-part of the request",
        "data1": "3.141529",
        "data_2": "2.71828",
        "and so on...": "any data the device has sent to the server",
        ...
    }
```

Set device status

0.2 POST /status/{device}

Request *raw*

```
Content-Type: application/json
```

```
{
    "data1": "3.141529",
    "data_2": "2.71828",
```

```
    "and so on...": "any data here will be stored by the server",
    ...
}
```

Response

```
200 (OK)
Content-Type: application/json
```

```
    Will return the same as a GET request to [/command/{device}]
```

Manage sensor data for a single sensor

Resource for storing and fetching sensor data for a given sensor for a given device.

Get sensor data

0.3 GET /data/{device}/{sensor}

Request *raw*

Response

```
200 (OK)
Content-Type: application/json
```

```
    {
        "timestamp": "Timestamp in milliseconds when the server received the last status
update",
        "_id": "Database id, not needed for anything",
        "device_id": "The same as the {device}-part of the request",
        "sensor": "The same as the {sensor}-part of the request",
        "any_key": "data specified by the device when updating the sensor data",
        ...
    }
```

Set sensor data

0.4 POST /data/{device}/{sensor}

Request *raw*

```
Content-Type: application/json
```

```
{
    "any_key": "data specified by the device when updating the sensor data",
    ...
}
```

Response

```
200 (OK)
Content-Type: application/json
```

```
    Will return the same as a GET request to [/command/{device}]
```

Manage sensor data for multiple sensors

Resource for storing and fetching sensor data for all sensors for a given device.

Get the data from all the device's sensors

0.5 GET /data/{device}

Request *raw*

Response

```
200 (OK)
Content-Type: application/json
```

```
    [
        {
            "timestamp": "Timestamp in milliseconds when the server received the last status
update",
            "_id": "Database id, not needed for anything",
            "device_id": "The same as the {device}-part of the request",
            "sensor": "The id of this sensor",
            "any_key": "data specified by the device when updating the sensor data",
            ...
        },
        {
            "timestamp": "Timestamp in milliseconds when the server received the last status
update",
            "_id": "Database id, not needed for anything",
            "device_id": "The same as the {device}-part of the request",
            "sensor": "The id of this sensor",
            "any_key": "data specified by the device when updating the sensor data",
            ...
        },
        ...
    ]
```

Set the data for several of the device's sensors

0.6 POST /data/{device}

Request *raw*

```
Content-Type: application/json
```

```
[
    {
        "sensor": "The id of this sensor",
        ...
    },
    {
        "sensor": "The id of this sensor",
        ...
    },
    ...
]
```

Response

```
200 (OK)
Content-Type: application/json
```

```
    Will return the same as a GET request to [/command/{device}]
```

Manage a device's command queue

Resource for adding commands to a device's command queue and retrieving the command queue.

Get the device's command queue and flush it

0.7 GET /command/{device}

Request *raw*

Response

```
200 (OK)
Content-Type: application/json
```

```
    [
        {
            "timestamp": "Timestamp in milliseconds when the server received the last status
update",
            "any_key": "Any data can go here",
            ...
        },
        ...
    ]
```

Add a command to the device's command queue

0.8 POST /command/{device}

Request *raw*

```
Content-Type: application/json
```

```
{
    "any_key": "Any data can go here",
    ...
}
```

Response

```
200 (OK)
Content-Type: application/json
```

```
    {}
```