Hardware API v2.0.1

Common provisions

Terminology

The terminology of RFC 2119 (specifically **must**, **should**, **may** and their negatives) applies. The word **will**, when applied to the Hardware Service API ("the API"), has the same meaning as **must**.

Protocol

The API supports communication over HTTPS only.

Encoding

The API supports communication using JSON encoding only. The client **must** submit the headers `Content-Type: application/json` and `Accept: application/json` for all requests. Failure to do so **will** result in a `415 Unsupported Media Type` response. The API **will** include the header `Content-Type: application/json` with its response.

Authentication

Unless otherwise specified, the endpoints in the API are authenticated by a JWT bearer token. Three token sources are accepted:

- Tokens generated by Amazon Cognito and acquired from the `accessory/login` endpoint;
- Tokens generated by Amazon Cognito and acquired as part of authentication to the Users Service;
- Tokens returned from the Website service's `/token` endpoint as the value of the `access_token` property. See the documentation at /Website/README.md#Oauth.

The client **must** submit the header `Authorization: <JWT>` with all requests. Failure to do so, or submitting an invalid or expired JWT, **will** result in a `401 Unauthorized` response.

General responses

In addition to the AWS API Gateway responses and the specific responses for each endpoint, the server **may** respond with one of the following HTTP responses:

- `400 Bad Request` with `Status` header equal to `InvalidSchema`, if the JSON body of the request does not match the requirements of the endpoint.
- `404 Unknown` with `Status` header equal to `UnknownEndpoint`, if an invalid endpoint was requested.

Schema

Simple

The following simple types **may** be used in responses:

- `string`, `number`: as defined in the JSON Schema standard.
- `Uuid`: a `string` matching the regular expression `^[0-9a-f]{8}-[0-9a-f]{4}-[0-9a-f]{4}-[0-9a-f]{4}-[0-9a-f]{12}$`, that is, the string representation of an RFC 4122 UUID.
- `Datetime`: a `string` matching the regular expression `/\d{4}-\d{2}-\d{2}T\d{2}:\d{2}:\d{2}(Z|+\d{2}:\d{2})/` and representing a date and time in full ISO 8601 format.
- `DeviceType`: one of the `strings` `accessory`, `hip` or `ankle`.
- `MacAddress`: a `string` matching the regular expression `/{[0-9a-f]{2}(:[0-9a-f]{2}){5}/`, that is six groups of two hexadecimal characters, separated by colons.
- `VersionNumber`: a `string` matching the regular expression `/\d+\.\d+(\.\d+)?/`, that is a Semantic Versioning version number (in either `MAJOR.MINOR.PATCH` or `MAJOR.MINOR` format)

Accessory

An `Accessory` object **must** have the following schema:

```
{
    "battery_level": Number,
    "bluetooth_name": String,
    "firmware_version": VersionNumber,
    "mac_address": MacAddress,
    "memory_level": Number,
    "state": String
}
```

The following constraints **will** apply:

- `battery_level` and `memory_level` **must** be a number between 0 and 1 inclusive.

Sensor

A `Sensor` object **must** have the following schema:

```
{
    "battery_level": Number,
    "firmware_version": VersionNumber,
    "gyro_offset": [ Number, Number, Number]
    "mac_address": MacAddress,
    "memory_level": Number,
}
```

The following constraints **will** apply:

- `battery_level` and `memory_level` **must** be a number between 0 and 1 inclusive.

- `gyro_offset` **must** be a list of exactly three Numbers.

Firmware

A `Firmware` object **must** have the following schema:

```
{
    "device_type": string,
    "version": VersionNumber
}
```

The following constraints **will** apply:

- `device_type` **will** be one of the strings `accessory`, `ankle` or `hip`.

Endpoints

Accessory

Register

This endpoint can be called to register a new accessory.

Query String

The client **must** submit a request to the endpoint `/accessory/{mac_address}/register`, where `mac_address` **must** be a MacAddress. It **should** correspond to the MAC Address of the accessory.

Request

The client **must** submit a request body containing a JSON object with the following schema:

```
{
    "password": String,
    "hardware_model": String,
    "firmware_version": VersionNumber,
    "settings_key": String
}
```

- `password` **must** be a string containing 8 or more characters, with no leading or trailing spaces.
- `hardware_model` **must** be a string of between 1 and 256 characters. It **should** uniquely identify the hardware model of the accessory. This value is immutable.
- `firmware_version` **must** be a VersionNumber, which **should** identify the version of the firmware installed on the accessory.
- `settings_key` **must** be a string of between 1 and 256 characters.

```
POST /hardware/2_0/accessory/1d:3a:42:5d:g5:ea/register HTTP/1.1
Host: apis.env.fathomai.com
```

```
Content-Type: application/json

{
    "password": "ffqkjhrqdkha2",
    "hardwareModel": "Model T",
    "firmwareVersion": "1.0",
    "settingsKey": "123456"
}
```

Authentication is not required for this endpoint.

Responses

If the registration was successful, the Service **will** respond with HTTP Status `201 Created`.

If the request was not successful, the Service **may** respond with:

- `409 Conflict` with `Status` header equal to `DuplicateEntity`, if an accessory with that MAC address has already been registered.

Login

This endpoint can be called by an accessory, once registered, to acquire credentials with which to access other endpoints. The accessory **must** have been registered via a call to /accessory/{mac_address}/register prior to requesting this endpoint.

Query String

The client **must** submit a request to the endpoint `/accessory/{mac_address}/login`, where `mac_address` **must** be a MacAddress. It **should** correspond to the MAC Address of the accessory.

Request

The client **must** submit a request body containing a JSON object with the following schema:

```
{
    "password": String
}
```

- `password` **must** be a string containing 8 or more characters, with no leading or trailing spaces.

```
POST /hardware/2_0/accessory/1d:3a:42:5d:g5:ea/login HTTP/1.1
Host: apis.env.fathomai.com
Content-Type: application/json

{
    "password": "ffqkjhrqdkha2"
}
```

Authentication is not required for this endpoint.

Responses

If the authentication was successful, the Service **will** respond with HTTP Status `200 OK`, and with a body with the following syntax:

```
{
    "authorization": {
        "expires": String,
        "jwt": String
    },
    "mac_address": MacAddress
}
```

- `authorization.jwt` **will** be a String forming a valid JWT Bearer Token.
- `authorization.expires` **will** be a Datetime, representing the time at which the JWT will expire.
- `mac_address` **will** be the same MacAddress as submitted in the request.

Example response:

```
{
    "authorization": {
        "expires": "2018-02-19T18:31:19Z",
        "jwt": "eyJraWQ...ajBc4VQ"
    },
    "mac_address": "1d:3a:42:5d:g5:ea"
}
```

Sync

This endpoint can be called by an accessory to record an update to its state. The accessory **must** have been registered via a call to /accessory/{mac_address}/register prior to requesting this endpoint.

Query String

The client **must** submit a request to the endpoint `/accessory/{mac_address}/sync`, where `mac_address` **must** be a MacAddress. It **should** correspond to the MAC Address of the accessory.

Request

The client **must** submit a request body containing a JSON object with the following schema:

```
{
    "event_date": Datetime,
```

```
    "accessory": Accessory,
    "sensors": [ Sensor, Sensor, Sensor ]
}
```

The `sensors` field **should** have exactly three elements.

Example request:

```
POST /hardware/2_0/accessory/1d:3a:42:5d:g5:ea/sync HTTP/1.1
Host: apis.env.fathomai.com
Content-Type: application/json
Authorization: eyJraWQ...ajBc4VQ

{
    "event_date": "2016-12-09T08:21:15Z",
    "accessory": {
      "state": "0x01",
      "battery_level": 0.89,
      "memory_level": 0.89,
      "firmware_version": "2.3.2",
      "bluetooth_name": "athl1"
    },
    "sensors": [
        {
            "mac_address": "aa:bb:cc:dd:ee:ff",
            "battery_level": 0.57,
            "memory_level": 0.57,
            "firmware_version": "1.2",
            "gyro_offset": [
              0.572344,
              0.572344,
              0.572344
            ]
        },
        ...
    ]
}
```

Responses

If the request was successful, the Service **will** respond with HTTP Status `200 OK`, and with a body with the following syntax:

```
{
    "accessory": Accessory,
    "sensor": Sensor,
    "latest_firmware": {
        DeviceType: Firmware,
        ...
    }
}
```

Example response:


```
{
    "accessory": {
        "id":"1d:3a:42:5d:g5:ea",
        "state":"0x01",
        "battery_level":0.89,
        "memory_level":0.89,
        "firmware_version":"2.3.2",
        "bluetooth_name":"athl1"
    },
    "sensors": [
        {
            "mac_address":"aa:bb:cc:dd:ee:ff",
            "battery_level":0.57,
            "memory_level":0.57,
            "firmware_version":"1.2",
            "gyro_offset":[
                0.572344,
                0.572344,
                0.572344
            ]
        },
        {
            "mac_address":"aa:bb:cc:dd:ee:ff",
            "battery_level":0.57,
            "memory_level":0.57,
            "firmware_version":"1.2",
            "gyro_offset":[
                0.572344,
                0.572344,
                0.572344
            ]
        },
        {
            "mac_address":"aa:bb:cc:dd:ee:ff",
            "battery_level":0.57,
            "memory_level":0.57,
            "firmware_version":"1.2",
            "gyro_offset":[
                0.572344,
                0.572344,
                0.572344
            ]
        }
    ],
    "latest_firmware": {
        "accessory": {
            "device_type": "accessory",
            "version": "1.1",
            "created_date": "2018-02-23T19:34:00Z"
        },
        "hip": {
            "device_type": "hip",
```

```
            "version": "1.0",
            "created_date": "2018-02-23T19:33:00Z"
        },
        "ankle": {
            "device_type": "ankle",
            "version": "1.2",
            "created_date": "2018-02-23T19:33:00Z"
        }
    }
}
```

Sensor

Patch

This endpoint can be called to register a new sensor, or update an existing one.

Query String

The client **must** submit a request to the endpoint `/sensor/{mac_address}`, where `mac_address` **must** be a MacAddress. It **should** correspond to the MAC Address of the sensor. The HTTP method **must** be `PATCH`. The `Content-Type` header **should** be `application/merge-patch+json`, as the request complies with RFC 7396.

Request

The client **must** submit a request body containing a JSON object with the following schema:

```
Sensor
```

With the following constraints:

- `battery_level` and `memory_level` **must** be a number between 0 and 1 inclusive.
- The client **may** not include all of the above fields in the request, but **should** include at least one.

```
PATCH /hardware/2_0/sensor/1d:3a:42:5d:g5:ea HTTP/1.1
Host: apis.env.fathomai.com
Content-Type: application/merge-patch+json
Authorization: ...

{
  "mac_address": "1d:3a:42:5d:g5:ea",
  "firmware_version": "1.2",
  "memory_level": 0.2
}
```

Responses

If the request was successful, the Service **will** respond with HTTP Status `200 Updated` or `201 Created`.

If the request was not successful, the Service **may** respond with:

- `409 Conflict` with `Status` header equal to `DuplicateEntity`, if a sensor with that MAC address has already been registered.

Multi-Patch

This endpoint can be called to register or update multiple sensors.

Query String

The client **must** submit a request to the endpoint `/sensor`. The `Content-Type` header **should** be `application/merge-patch+json`, as the request complies with RFC 7396.

Request

The client **must** submit a request body containing a JSON object with the following schema:

```
{
    "sensors": [
        Sensor,
        ...
    ]
}
```

The `sensors` field **should** contain at least one element.

```
PATCH /hardware/2_0/sensor HTTP/1.1
Host: apis.env.fathomai.com
Content-Type: application/merge-patch+json
Authorization: ...

{
    "sensors": [
        {
            "mac_address": "AB:CD:EF:12:34:56",
            "firmware_version": "1.2"
        },
        {
            "mac_address": "65:43:21:FE:DC:BA",
            "firmware_version": "1.4"
        }
    ]
}
```

Responses

If the request was successful, the Service **will** respond with HTTP Status `200 Updated` or `201 Created`.

If the request was not successful, the Service **may** respond with:

- `409 Conflict` with `Status` header equal to `DuplicateEntity`, if a sensor with that MAC address has already been registered.

Firmware

Get

This endpoint allows the client to get information about a firmware version, or determine the most recent available firmware.

Query String

The client **must** submit a request to the endpoint `/firmware/{device_type}/{version_number}`, where:

- `device_type` **must** be a `DeviceType`;
- `version_number` **must** be either a `VersionNumber` or the string `"latest"`

The request method **must** be `GET`.

Request

This method takes no request body.

Example request:

```
GET /hardware/2_0/firmware/accessory/latest HTTP/1.1
Host: apis.env.fathomai.com
Content-Type: application/json
```

Authentication is not required for this endpoint.

Response

The Service **will** respond either with an HTTP status of `200 OK` and a body with the following syntax:

```
{
    "firmware": Firmware
}
```

or, with an HTTP status of `303 See Other`, and a `Location` header pointing to another resource which **will** respond to the same request with a body matching the above schema.

If the `version_number` in the request was set to `"latest"`, the Firmware object returned **will** be the most recently-released firmware version for the requested device type.

Download

This endpoint allows the client to download the binary file for a given firmware version.

Query String

The client **must** submit a request to the endpoint `/firmware/{device_type}/{version_number}/download`, where:

- `device_type` **must** be a `DeviceType`;
- `version_number` **must** be either a VersionNumber or the string `"latest"`

The request method **must** be `GET`.

The client **must** submit an `Accept` HTTP header with value `application/octet-stream` in order to receive a raw binary response. Failure to supply this will result in the response being base-64 encoded.

Request

This method takes no request body.

Example request:

```
GET /hardware/2_0/firmware/accessory/latest/download HTTP/1.1
Host: apis.env.fathomai.com
Content-Type: application/json
```

Authentication is not required for this endpoint.

Response

The Service **will** respond with an HTTP status of `200 OK` and a body containing raw binary data.

Miscellaneous

Current time

This endpoint returns the current time.

Query String

The client **must** submit a request to the endpoint `/misc/time`.

Request

This method takes no request body.

Example request:

```
GET /hardware/2_0/misc/time HTTP/1.1
Host: apis.env.fathomai.com
Content-Type: application/json
```

Authentication is not required for this endpoint.

Response

The Service **will** respond with an HTTP status of `200 OK` and a body with the following syntax:

```
{
    "current_date": Datetime
}
```