Objenious
Receiving messages from devices

Version 1.0.1

1 Introduction

This document describes the interface definition between the Objenious platform and an external client application.

Messages from devices are forwarded to the external application using the HTTP/HTTPS protocol, with a JSON body.

The body can have the following formats:

- Values (only if messages are decoded by the Objenious platform)
- Messages.

2 Authentication

Should the external client application require authentication, the following can be setup within the Objenious portal:

- Request parameters located in the URL,
- Custom HTTP headers (e.g. API Token, Basic Authentication, ...).

3 Values format

```
{
    timestamp: "2016-05-04T16:01:26.572226000Z",
    device_id: 3421, // id on the objenious platform
    data: {
        Temperature: 21,
        ButtonPushed: true
    },
    device_properties: {
        external_id: "...", // id on the client platform
        deveui: "...",
        appeui: "...",
        property: "value"   // client defined device properties
    },
    lat: 48.64332, // latitude
    lng: 2.34123, // longitude
    geolocation_type: "network" // tdoa, network, device or fixed
}
```

Notes:

- When using the values format, a HTTP request will only contain values measured at a specific time. If the device sends a message containing multiple measurements within a single payload, as many HTTP requests will be sent as there are measurements.

4 Message format

```
{
    id: "d5e2c4cc722-126c17ed", // id of message
    timestamp: "2016-05-04T16:01:26.572226000Z
    device_id: 3421, // id on the objenious platform
    type: "uplink", // join, uplink, downlink, external
    count: 21, // fcount
    payload_encrypted: "...",
    payload_cleartext: "...",
    payload: [{
        timestamp: "2016-05-04T16:01:26.572226000Z",
        data: {
                Temperature: 21,
                ButtonPushed: true
        }
    }],
    device_properties: {
        external_id: "...", // id on the client platform
        deveui: "...",
        appeui: "...",
        property: "value"   // client defined device properties
    },
    lat: 48.64332, // latitude
    lng: 2.34123, // longitude
    geolocation_type: "network", // tdoa, network, device or fixed
    command_id: 456, // as set on the downlink request (downlink)
    error: "...", // error (downlink/join)
    delivered_at: "2016-05-04T16:01:26.572226000Z" // (downlink)
}
```

Notes:

- payload_cleartext is only present if decryption is configured on the Objenious platform,
- payload is only present if decryption and decoding are configured on the Objenious platform, and will be found in uplink or external messages,
- payload may contain multiple entries, as decoded,
- join messages are sent to your platform when a device joins the network,
- downlink messages are sent to your platform when a downlink request is delivered to the device or when the network decides that a downlink cannot be sent, either due to a timeout, a network error or a message formatting problem,
- external messages are messages received by the Objenious platform from external platforms – typically a third-party service used to decode uplink payloads.

5 Formats

The following formats are used:

- Times: RFC3339 with nanoseconds, UTC timezone
- DevEUIs and AppEUIs: bigendian/hexadecimal
- Latitudes and longitudes: degrees
- Payloads (encrypted and cleartext): hexa
- External_id: string

6 Error handling

If a message cannot be sent to a remote application due to an error (connectivity problem, unavailable remote server, …), the Objenious will try again using the following strategy:

- When the remote server reports a client error (HTTP status codes 4XX), no retries will be attempted,
- When the remote server reports a server error (HTTP status codes 5XX) or when the remote server is unreachable, retries will be attempted using an exponential backoff algorithm,
- If the remote server is still unreachable, the message will be requeued for later redelivery, and redeliveries will be attempted for the next 7 days.

When retries are attempted, the ordering of message is not guaranteed.