Live Objects

Orange

User manual
# Table of content

1. **INTRODUCTION** .................................................................................................................. 5
   1.1. Context .............................................................................................................................. 5
   1.2. Purpose of this document ................................................................................................. 5
   1.3. References ........................................................................................................................ 5

2. **LIVE OBJECTS PORTAL USER MANUAL** .................................................................. 6
   2.1. Access ............................................................................................................................... 6
       2.1.1. Get a Live Objects account ....................................................................................... 6
       2.1.2. Credentials ............................................................................................................... 6
       2.1.3. Log into Live Objects portal .................................................................................... 7
           2.1.3.1. Supported browsers ......................................................................................... 7
           2.1.3.2. Login ............................................................................................................... 7
   2.2. Organization of the portal .................................................................................................. 8
       2.2.1. Header ....................................................................................................................... 8
   2.3. Dashboard .......................................................................................................................... 10
       2.3.1. Total number of devices per connectivity ................................................................. 11
       2.3.2. Quotas ...................................................................................................................... 11
       2.3.3. Fleet activity ........................................................................................................... 12
       2.3.4. Detailed activity ..................................................................................................... 13
       2.3.5. Custom dashboards ................................................................................................ 13
           2.3.5.1. Creation of a widget “value” ............................................................................ 15
           2.3.5.2. Creation of a widget “Line chart” ..................................................................... 16
           2.3.5.3. Creation of a widget “map” ............................................................................. 17
           2.3.5.4. Creation of a widget “jauge” ............................................................................ 17
           2.3.5.5. Creation of a widget “bar chart” ...................................................................... 18
           2.3.5.6. Creation of a widget “On/Off” ........................................................................ 19
           2.3.5.7. Creation of a widget “Statistics” .................................................................... 20
   2.4. Device management .......................................................................................................... 21
       2.4.1. Display the fleet per connectivity ............................................................................. 22
       2.4.2. Devices statuses ...................................................................................................... 22
       2.4.3. Command status ..................................................................................................... 24
       2.4.4. Interfaces and device properties .............................................................................. 26
       2.4.5. Mass import of devices .......................................................................................... 27
       2.4.6. Export a list of devices ............................................................................................ 31
       2.4.7. LoRa devices ........................................................................................................... 32
           2.4.7.1. Add a LoRa device .......................................................................................... 32
           2.4.7.2. Deactivate and reactivate a LoRa device ......................................................... 34
           2.4.7.3. Delete a LoRa device ....................................................................................... 35
           2.4.7.4. Details of a LoRa device ................................................................................... 35
           2.4.7.5. Modify a LoRa device ...................................................................................... 38
           2.4.7.6. Add a connectivity option ............................................................................... 39
           2.4.7.7. See data issued by a LoRa device (uplink) .................................................... 41
           2.4.7.8. Send a command to a LoRa device (downlink) ............................................ 43
           2.4.7.9. LoRa gateways ............................................................................................... 45
2.4.8. MQTT devices .......................................................................................................................... 48
  2.4.8.1. Add an MQTT device ........................................................................................................ 48
  2.4.8.2. Delete an MQTT device .................................................................................................... 49
  2.4.8.3. Details of an MQTT device ............................................................................................... 49
  2.4.8.4. Update parameters of an MQTT device ........................................................................... 51
  2.4.8.5. Send a command to an MQTT device .............................................................................. 54
  2.4.8.6. Update the firmware of an MQTT device ......................................................................... 56
2.4.9. SMS devices ............................................................................................................................. 59
  2.4.9.1. Add an SMS device ......................................................................................................... 60
  2.4.9.2. Delete an SMS device .................................................................................................... 61
  2.4.9.3. Send a command to an SMS device ............................................................................... 61
  2.4.9.4. Modify an SMS device ................................................................................................... 63
2.4.10. External connector .................................................................................................................. 63
2.4.11. Groups .................................................................................................................................... 64
2.4.12. Campaign management .......................................................................................................... 66
  2.4.12.1. Create a campaign ........................................................................................................ 66
  2.4.12.2. Follow a campaign ........................................................................................................ 69
  2.4.12.3. Cancel a campaign ....................................................................................................... 70
  2.4.12.4. Delete a campaign ........................................................................................................ 70
  2.4.12.5. Status of a campaign .................................................................................................... 71
  2.4.12.6. Status of a target device ............................................................................................... 71
  2.4.12.7. Campaign report .......................................................................................................... 72
2.5. Data .............................................................................................................................................. 73
  2.5.1. Data export ........................................................................................................................... 76
  2.5.2. Advanced visualization (Kibana) ......................................................................................... 76
  2.5.3. Messages queue (fifo) ....................................................................................................... 78
  2.5.4. Routing data and events ....................................................................................................... 80
    2.5.4.1. Create a routing rule ...................................................................................................... 80
    2.5.4.2. Routing to fifo .............................................................................................................. 82
    2.5.4.3. Push http ...................................................................................................................... 83
    2.5.4.4. Suspend/Enable a rule .................................................................................................. 85
  2.5.5. Data consumption by applications ....................................................................................... 85
    2.5.5.1. Explore data .................................................................................................................. 86
    2.5.5.2. Consume data in real time ............................................................................................ 86
      2.5.5.2.1. Consume data with MQTT ..................................................................................... 86
      2.5.5.2.2. Consume data with push HTTP ............................................................................. 87
  2.5.6. Manage decoders .................................................................................................................... 87
    2.5.6.1. Type of decoders .......................................................................................................... 87
    2.5.6.2. Visibility of decoders .................................................................................................... 88
    2.5.6.3. Data model and indexing ............................................................................................... 88
    2.5.6.4. Create a binary decoder (configurable) ........................................................................ 90
    2.5.6.5. Create a CSV decoder (configurable) .......................................................................... 94
    2.5.6.6. Create a programmable decoder ................................................................................ 95
    2.5.6.7. Disable/enable a decoder ............................................................................................. 95
    2.5.6.8. Assign a decoder to a device ......................................................................................... 96
  2.5.7. Activity Logs .......................................................................................................................... 96
    2.5.7.1. LoRa connectivity ......................................................................................................... 97
    2.5.7.2. MQTT connectivity ...................................................................................................... 97
    2.5.7.3. Storage ......................................................................................................................... 98
    2.5.7.4. Notification .................................................................................................................. 98
    2.5.7.5. Limitations .................................................................................................................. 98
2.6. Configuration ............................................................................................................................... 98
  2.6.1. Identification of the account ............................................................................................... 98
2.6.2. Identification of the current user ................................................................. 99
2.6.3. Traffic limits ................................................................................................. 99
2.6.4. Roles and rights in Live Objects .................................................................. 100
2.6.5. Manage users ............................................................................................... 102
  2.6.5.1. Create a user .......................................................................................... 103
  2.6.5.2. Suspend / reactivate a user ................................................................. 104
  2.6.5.3. Delete a user ......................................................................................... 104
2.6.6. Manage API keys .......................................................................................... 104
  2.6.6.1. Create an API key ............................................................................. 105
  2.6.6.2. Hierarchy of API Keys ...................................................................... 107
  2.6.6.3. Regenerate an API key ..................................................................... 109
  2.6.6.4. Disable / enable an API key .............................................................. 109
  2.6.6.5. Delete an API key ............................................................................. 109
  2.6.6.6. Put an API key in debug mode ........................................................... 109
  2.6.6.7. Associate a certification authority ...................................................... 110
2.6.7. Manage firmwares .......................................................................................... 111
2.7. Alarms ............................................................................................................. 114
  2.7.1. Display alarms ......................................................................................... 115
  2.7.2. Set a silent machine rule ....................................................................... 115
3. RESOURCES .................................................................................................... 118
1. Introduction

1.1. Context

Live Objects is software as a service (SaaS) who provides a set of tools for Machine To Machine (M2M) and Internet of things (IoT).

Main features provided by Live Object are:

- Advanced management of connected devices (provisioning, supervision, configuration, firmware update, campaign…)
- Application programming interfaces (API) to connect devices and business applications, and manage all Live Objects features
- Message routing
- Data management (event/state processing, dataviz, search, data message enrichment)
- Data storage

**Important:** Depending on subscribed offer (LoRa, Discover, Express, Advanced, Premium) some features may be available or not. In this document all features are presented.

1.2. Purpose of this document

This user manual is a support to the use of Orange Live Objects portal. Some features only available by API are not described here, but in the developer guide referenced in the table below.

1.3. References

<table>
<thead>
<tr>
<th>N°</th>
<th>title</th>
<th>reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Orange documents</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Live Objects complete training</td>
<td><a href="https://www.slideshare.net/EricCombe/live-objects-concepts-and-apis-training-129691537">https://www.slideshare.net/EricCombe/live-objects-concepts-and-apis-training-129691537</a></td>
</tr>
<tr>
<td>3</td>
<td>Swagger (API reference)</td>
<td><a href="https://liveobjects.orange-business.com/#/swagger">https://liveobjects.orange-business.com/#/swagger</a></td>
</tr>
</tbody>
</table>
2. Live Objects portal user manual

2.1. Access

2.1.1. Get a Live Objects account

You can get a trial account (Discover) online at [https://liveobjects.orange-business.com/#/request_account](https://liveobjects.orange-business.com/#/request_account).

If you want an account with LoRa connectivity included, check the corresponding checkbox. In this case the creation of the account is not immediate, and you will be notified by email when ready.

Then, you can contact a commercial interlocutor at liveobjects.contact@orange.com

2.1.2. Credentials

The creation of your Live Objects account triggers an activation email. This email contains an activation link valid for 7 days. When you will activate your account you will have to set your password.
The password must contain 8 characters with uppercase, lowercase, numbers and special characters (/^-+.?;:,!%û$@\').

Then a new email is issued to confirm your account creation. It contains the identifier you will use to log into Live Objects portal.

2.1.3. Log into Live Objects portal

2.1.3.1. Supported browsers

Supported browsers are:
- Firefox 38 and over
- IE 11,
- Edge 13 and over
- Chrome 49 and over

2.1.3.2. Login

Go to https://liveobjects.orange-business.com

Connect with the identifier you have received in your welcome email, and the password you set during the activation process. If you don’t remember it, use the link “forgotten password?”
To reset your password type your login into the form, then an email is issued to the email address of the account. This email contains a reset link valid only for 10 minutes.

### 2.2. Organization of the portal

#### 2.2.1. Header

The header allows diving into the main areas of Live Objects portal:
- «Dashboard» is the welcome page, which gives a synthetic view of the account activity
- «Devices» to manage the fleet of devices
- «Data» to access the data stored into Live Objects, manage fifo and routing rules, and activity logs
- «Configuration» for Live Objects account configuration (API keys, users, decoders…)
- «Alarms» to manage alarms

The “help center” provides access to various resources and support, among them
- The API documentation (swagger)
- The user manual (this document)
- The developer guide

One link to log-off, and to get to your account information's:
- Account Ids (to be communicated when calling to customer support)
- The current user connected
o Name of the account, date of creation and email of the principal user (used for the creation of the Live Objects account)

Depending on your country you can switch at any moment from English to your native language. Live Objects refresh automatically
2.3. Dashboard

The dashboard is the welcome page of the portal. It gives a synthetic view of the account activity.
2.3.1. Total number of devices per connectivity

This view gives the number of devices currently managed per connectivity.
Note that the same device can have multiple interfaces if it uses multiple connectivities to communicate (for example a LoRa interface and an SMS interface)

2.3.2. Quotas

These quotas depend on the offer that has been subscribed.
- Message queues : Number of queues created / Max number of queues allowed
- API keys: Number of key created / Max number of key allowed
- Users: Number of users created / Max number of users allowed
2.3.3. Fleet activity

This view allows selecting for a given period (last 7 days, last 4 weeks, or last 12 months) the global activity of the Live Objects account:

- number of devices that have communicated over the period (history available for 7 days or last 12 months)
- traffic in number of messages (in/out) or in volume (bytes) (history available for 7 days or last 12 months or last 4 weeks)

« incoming messages» are any messages entering into the platform. Examples:

- uplink messages from device to the platform
- a business application on top of Live Objects which send REST API requests (POST, PUT, PATCH) to the platform (ex: search request, send command …)

« outgoing messages» are any messages issued by the platform to the outside. Examples:

- A downlink command from the platform to a device
- Answers to REST API call (GET)
- Consumption of messages by MQTT
2.3.4. Detailed activity

A synthesis of the activity is also available per connectivity

For each connectivity:
- The number of devices currently managed
- Statistics over the selected period. Note that on this period, the number of devices could vary from the current state of the fleet, depending on additions and deletions that may have occurred in the meantime.

Some information are specific to connectivity: devices Connected/disconnected for MQTT vs activated/deactivated for LoRa

The number of outgoing mqtt messages includes all mqtt messages coming out of the platform whether they are destined for devices (device management) or applications (data consumption).

2.3.5. Custom dashboards

You can build your own dashboards to visualize your data. A dashboard consists of graphical objects called "widgets" which allow visualizing the data in different ways.

Notes:
- Custom dashboards are user specific. They are not shared to other users of the account.
- You can set an automatic refresh of a dashboard. In this case each widget is refreshed sequentially according to the defined delay. However, it is still possible to refresh a widget individually by clicking on its "refresh" icon.
• To visualize data sent by LoRa devices, it is necessary that the hexadecimal payload carrying these data be decoded (see paragraph "Manage decoders"). Network metadata (snr, rssi, sf, esp) can still be accessed without decoding.

• Only the most recent 10,000 data are displayed by a widget. If a series has more data on the selected period, this is indicated by a yellow icon next to the name of the series.
2.3.5.1. Creation of a widget “value”

This widget is used to display the last value of a data sent by a device. To create a “value” widget, click on the corresponding button and select the device that transmits the data to be viewed.

Once the device has been selected, the “Data to display” field presents the structure of the message sent by this device. Select the field carrying the data to display:
2.3.5.2. Creation of a widget “Line chart”

This widget makes it possible to display in the form of a curve or a cloud of points the values taken by a data reported by a device over a period of time. To create a “Line chart” widget, click on the corresponding button and select the device that transmits the data to be viewed.

Once the device has been selected, the “Data to display” field presents the structure of the message sent by this device. Select the field carrying the data to display:

NB: A widget line chart can only display numeric values
2.3.5.3. Creation of a widget “map”

This widget makes it possible to display on a map the geographical position of the device as well as the last value of data transmitted by this device. To create a map widget, click on the corresponding button, then select the device that issues the data to be viewed in the same way as for previous widgets.

Once the device has been selected, the “Data to display” field presents the structure of the message sent by this device. **Warning:** You do not have to go for the location information that will automatically be shown on the map. The data you have to select is the one you want to display on the map where your device is located. For example the temperature indicated by the device during its last geolocation.

2.3.5.4. Creation of a widget “jauge”

The gauge is used to represent the last value reported by a device in the form of a gauge on which you can include a minimum value and a maximum value. These min and max are used for presentation only, a device will be able to report values under or above these limits, the gauge appears in this case empty or full.
2.3.5.5. Creation of a widget “bar chart”

With a bar chart widget you have the possibility to visualize the compared evolution over the of a value reported by one or several devices.

As for other widgets, you must select the data to display once the devices are selected. The "Data to display" field contains the structure of the data reported by these devices.

The selected approximation method corresponding to your use case, for example:
- Monitoring the average / max / min temperature over the month. (select "average of measurements", "maximum of measurements", or "minimum of measurements")
- Cumulative precipitation volume over the month (select "sum of measurements")

NB 1: the y-axis is automatically set to the minimum value over the selected time period, unless the "stacked bars" box is checked in which case the scale starts at zero.

NB 2: A bar chart widget can only display numeric values
2.3.5.6. Creation of a widget “On/Off”

The widget “On/off” allows a simple conditional display
2.3.5.7. Creation of a widget “Statistics”

The statistics widget allows you to display a value based on simple aggregations. For example below, the value that will be displayed will be the average of the daily temperatures reported by the 3 selected devices, and if this value is greater than 30 °C it will be displayed in red.

If the option “The last measurement of each device” was selected, only the last value of each device would have been used to average the measurements, and not the daily average of each device.
2.4. Device management

You can manage your fleet of devices in the menu « Devices ». 
2.4.1. Display the fleet per connectivity

Because devices have characteristics proper to their connectivity, you have to first select the connectivity in the dropdown list (1).

You can filter the list of devices with several criteria. Click on the field « Add filters » to choose them (2).

You can select devices unitary, per page or totally, by using the checkbox. The number of selected devices / total number of devices is recalled in the top of the list (3).

You can make actions on the selection: export to a csv file, move from group, deletion (4).

The state of the device and its last communication and if necessary an alarm if it has no longer communicates for a period defined by a "silent machine" alarm (5).

2.4.2. Devices statuses

The status that the devices can take depends on their connectivity.

LoRa devices

- **Registered**: The device has been registered in the network with the parameters specified when it was created. Neither uplink data nor Join Request has yet been received by the platform.
- **Initializing**: The network received a Join Request from the device
- **Initialized**: The network sent a Join Accept to the device
- **Activated**: At least one uplink issued by the device was received by Live Objects (excluding MAC messages)
- **Deactivated**: The device has been deactivated in Live Objects. He can no longer communicate on the network (See the paragraph "Deactivate and reactivate a LoRa device"
- **Reactivated**: The device has been reactivated in Live Objects. Not being able to know a priori the state of the device at the time of the reactivation, the state will pass to "Initializing" if a Join Request is received, or directly to "Activated" if an uplink is received.
- **Error**: This status, rare, is displayed in case of problem of configuration of the equipment in the network. If this status appears, contact your support.
MQTT devices
- **Registered**: The device has been preprovisioned (by portal), but has never connect
- **Online**: The MQTT connection of the device is active
- **Offline**: The device has already connected at least once but its MQTT connection is inactive

SMS devices
- **Online**: The SMS device is activated at Live Objects level
- **Offline**: The SMS device is deactivated at Live Objects level

External connector device
- **Registered**: The device has been pre-provisioned but has never connect
- **Online**: The connector indicates that the device is connected and ready to communicate with the Live Objects platform *
- **Offline**: The connector indicates that the device is not connected and that it is therefore not ready to communicate with the Live Objects platform *

(* NB: In the case of the external connector, the state displayed in Live Objects is that reported by the external connector, and not by the end device itself. See the paragraph "External connector" for more information on how this works)

2.4.3. Command status

Commands or downlinks are the messages that you send to your devices from Live Objects.

The unified command engine of Live Objects allows you to finely track the progress of a command through its different states.

To track the status of a command sent to a device, first select the device from the "Devices" menu. Then in the device details select the "Commands (Downlink)" menu to view the history of the commands sent to this device. Finally, click on a command to display its details.
The first part of the screen shows the progress of the command.
Each type of connectivity (LoRa, SMS, MQTT, etc.) has its own specificities linked to its intrinsic mechanics. Live Objects’s generic command engine masks this complexity by managing commands seamlessly regardless of the connectivity used by the device to communicate.

So an order always has three states:
- **Pending**: The command is pending in Live Objects (for example a previous order is being executed, or the device is not available to receive a command)
- **Processing**: The command is being processed (being routed by the network)
- **Processed**: The command is finished. This means that it has been processed by Live Objects in accordance with the request (with or without acknowledgment)

These three generic states correspond to sub-states, the "delivery status" which provide more precise information on the command processing and whose meaning is specific to the connectivity used ("sending", "sent", "delivered"). These states are described in detail in the paragraphs corresponding to your type of device.

### 2.4.4. Interfaces and device properties

Devices can use different types of connectivity to communicate with Live Objects. The specificities related to these different connectivities are managed by "interfaces". A device can have multiple interfaces if it connects via different networks for example a LoRa interface and an SMS interface.

The information concerning the device is presented in two parts:
- Information independent of connectivity and common to all devices in Live Objects (detail below)
- Information specific to the network interfaces they use to connect to Live Objects (see specific paragraphs)
Name: Name of the device. Free.

Device ID: The unique device identifier in Live Objects (URN). It must be on the form `urn:lo:nsid:{namespace}:{id}` with namespace and id customizable.

Default StreamId: It allows to specify the stream in which the data emitted by the device will be stored. By default each device has its own stream, but it is quite possible to collect data from several devices in the same one.

Tags: Tags are free labels that you can attribute to your devices. You can use tags to ease sorting or identification in the fleet. You can enter up to 10 tags of 32 characters per device.

Group: Select here the group you want that your device belongs to. See paragraph « Groups »

Properties: You can define some properties for your devices. Properties allow you to enrich your device management. You can set up to 10 properties (up to 128 characters for the key, and up to 256 characters for the value).

2.4.5. Mass import of devices

This feature is currently only available for LoRa and SMS devices. Mass import of MQTT devices is planned in a future release.
To access this feature, go to the "Devices" menu and click on the "Mass import" button.

The import is in three steps:

The first one, optional, offers you a pre-formatted import file template with the parameters to be filled corresponding to the type of devices you want to import. The fields to be filled in are the same as those to be provided during unitary creation (see the corresponding paragraphs). You can skip this step if you already have your import file ready yet.
The second step is to import the file and verify that the format is correct.
Only the correction of all errors allow to pass to the next step

The third step is the actual creation of the devices contained in the import file. To confirm the mass creation click on the button “import the devices”
2.4.6. Export a list of devices

You can export the list of your devices in a csv file. For that go to the menu « Devices » then select the type of equipment (LoRa, MQTT or SMS). Then select the devices to export (all or a subset) and click on the icon « export ».
2.4.7. LoRa devices

2.4.7.1. Add a LoRa device

To add a LoRa device, click on the button “Add device”

The creation form opens with the parameters specific to a LoRa device.
**Name**: Name of the device

**Profile**: The « Device profile » contains some predefined network parameters as the « spreading factor » that the device will use for JOIN, Class and RX frame. Select in the list the profile corresponding to your device. If you don’t find the model corresponding to your device choose the generic profile Generic_classA_RX2SF12 (if your device is a class A) or Generic_classC_RX2SF12 (if your device is a class C)

**Decoder**: Decoders allow translating a payload from hexadecimal into data understandable by humans and indexable. Select in the list the decoder corresponding to your device. If there is no one, you can create your own in the menu “Configuration”. If the field “Decoder” doesn’t appear it means that the feature is not enabled to your account. See “manage decoders” paragraph.

**DevEUI**: The unique identifier of the device in the LoRa network. It is provided by the device vendor.

**Device ID**: The unique device identifier in Live Objects (URN). By default, it is built in the namespace « lora » with the devEUI as device identifier. You can custom the device ID with your own namespace instead of « lora » and with a device identifier different than devEUI

**AppEUI, AppKey**: These parameters belong to the owner of the device or the application used with it (See LoraWan specification)

**Default StreamId**: It allows to specify the stream in which the data emitted by the device will be stored. This field is automatically populated when you enter the devEUI, but it can be customized. By default each device has its own stream, but it is quite possible to collect data from several devices in the same one.

**Tags**: Tags are free labels that you can attribute to your devices. You can use tags to ease sorting or identification in the fleet. You can enter up to 10 tags of 32 characters per device.
Connectivity options: This field is displayed only if connectivity options have been activated on your Live Objects account. See paragraph « Add a connectivity option »

Group: Select here the group you want that your device belongs to. See paragraph « Groups »

Properties: You can define some properties for your devices. Properties allow you to enrich your device management.

Note: Orange recommends « Over the Air » activation (OTA) on its network. Only this activation mode is proposed in the Live Objects portal. If you need « activation by personalization » (ABP) please contact your customer support.

2.4.7.2. Deactivate and reactivate a LoRa device

It is possible to prevent one or more LoRa devices from communicating on the network temporarily without having to delete them. To do this, select the devices concerned by ticking the corresponding box in the list, and click on the "Disable" button.

To reactivate one or more devices, proceed in the same way by clicking on the "Activate" button.
2.4.7.3. Delete a LoRa device

It is possible to delete LoRa device from the platform. This action will also remove it from the Orange LoRa network. The data that have been issued by the device are not deleted and remain accessible in the menu “Data”.

To delete one or more devices, select them by ticking the corresponding box in the list, and click on the “Delete” icon.

2.4.7.4. Details of a LoRa device

To access detailed information about a LoRa device, click on the desired device in the main list of the “Park” menu.

The menu **Identity** is first displayed. It provides general information on the device and details of its connectivity interfaces.

- The last communication date is based on the timestamp field applied by the LoRa gateway that has forwarded the message
Fred Explorer Kit

Device information

- **Name**: Fred Explorer Kit
- **Device ID**: 00012345678901234567
- **Default stream ID**: 00012345678901234567
- **Group**: Demo
- **Tags**: Demo

Properties

- **Last update date**: 09/09/2019 5:34:47 PM
- **Registration date**: 09/09/2019 11:50:38 AM

Interface - LoRa

- **Status**: Activated
- **Profile**: GENERIC_GBEBA_RX2SF12
- **DevEui**: 00012345678901234567
- **AppEui**: 457675735857
- **AppKey**: ***************
- **Activation mode**: OTAA
- **Last comm.**: 09/09/2019 5:34:47 PM
- **Last update date**: 12/19/2019 12:36:09 PM
- **Creation date**: 09/08/2019 12:36:09 AM
- **Connectivity options**: Uplink Ack (deactivated)
  - GeoLocation TDOA (deactivated)
The menu **Monitoring** provides:

- the network signal: it is a quality indicator of the connectivity calculated by Live Objects based on various network parameters reported by the device (RSSI, SNR, PER ...)
- the battery level: It is based on the value transmitted to the network by the device (LoraWan standard)
- the geolocation of the device on a map with a circle showing the accuracy. The detail is available by clicking on:
  - date of the last location
  - GPS coordinates
2.4.7.5. Modify a LoRa device

To modify a LoRa device go to the details of the device and click on the “identity”.

Informations entered during the creation of the device are displayed. It is possible to modify the name, the profile, the decoder, the tags as well as the keys appEUI and appKey.

Changing the devEUI is not possible.

Note:
- The appKey is never displayed in clear.
- The name, and the appEUI, appKey keys are not editable if the device is disabled.
- The connectivity options must have been activated in your Live Objects account by your customer service to be selectable. See next paragraph.
2.4.7.6. Add a connectivity option

The connectivity options available are:

- TDOA geolocation: Under coverage conditions this type of geolocation allows a much greater accuracy than a standard geolocation
- "Ack Uplink": Allows the device to request an acknowledgment from the network for each uplink issued

(* Contact your customer service)

To get these options:

1) contact your customer service to activate them in your Live Objects account
2) then you’ll be able to add these options to your devices (See paragraph « Modify a LoRa device »)

To add a connectivity option to multiple devices at once, select your devices in the LoRa main view and click the edit icon.

In the window that opens check the desired option and select "ON" to enable the option on all selected devices or "OFF" to disable it. If an option is left unchecked, no modification concerning this option will be applied on the devices of the selection.

Examples below:
In addition, you can consult the details of the network parameters of the associated connectivity plans. The “connectivity plan” (CP) is the configuration plan for your account on the network. Most of the time you will not have to change the default CP, except for special needs and on the advice of your customer service.

In this example: We activate the option TDOA on all the devices of the selection.

In this example: We disable the option TDOA and activate the option Ack uplink on all the devices of the selection.
See data issued by a LoRa device (uplink)

To view the data reported by LoRa device go to the device details and click on "Uplink"
The chronological list of messages sent by the device is displayed along with the associated network information:

• The "Payload" column represents the application content sent by the device (no network message) in hexadecimal format. This content can be decoded by associating an appropriate decoder to the device.
• The "Decoding Status" column indicates whether a decoding was requested and whether it was correctly performed or not
• The "Gateway Number" column indicates the number of gateways that relayed the message (known also as "macro-diversity")
• Network information: Rssi, Snr, Esp, Sf corresponding to the gateway having relayed the message ("best LRR")

You can use the filter to search collected messages over a period of time.

A selector is set to 100 by defaults to limit the number of uplink to display in a page. The most recent 1000 can be displayed at most

This list can also be exported in a CSV file.
2.4.7.8. Send a command to a LoRa device (downlink)

To send a command to a LoRa device, go to the device details and click on the “Downlink”.

The history of commands sent to the device is displayed in a chronological order. The “status” column displays the status of the command and can have the following values:

- **Pending**: Means that the command is pending in Live Objects. It has not yet been transmitted to the network (for example because other previous orders are still being processed)
- **Sending**: The command has been transmitted to the network. It will be delivered to the device as soon as it connects or sends its next uplink
- **Sent**: The network sent the command to the device
- **Delivered**: A network acknowledgment has been received from the device
- **Error**: The command was not accepted by the network. It will therefore not be delivered to the device

The history of commands is exportable in a csv file.

To display the details of a command, click on it.
A command can be sent again without having to reenter it, for that, you just have to select it in the list and to click on the icon "Repeat".

To send a new order, click the button "Add a command"
The form requires entering two fields:
- the port used by the device (it must be greater than 0. Sending MAC commands is not possible via Live Objects)
- the command data (payload) in hexadecimal format (the validity of the format is checked)

It is possible to define the maximum duration during which the order can remain awaiting sending in Live Objects (status "pending") before being automatically canceled.

If the network acknowledgment level is selected, the command must be acknowledged by the device (ack bit set to true) in one of the next three uplinks consecutive to the downlink, otherwise the acknowledgment delay will be considered expired. You can ask Live Objects to try again automatically in this case using the "Failure Policy" option.

2.4.7.9. LoRa gateways
If you have your own LoRa gateways, you can manage them in your Live Objects portal, in the same place as your sensors. There are two prerequisites:
- These gateways must have been provisioned in the Orange network
- The option to view them must have been enabled on your Live Objects account
 -> For this please contact your customer service.

To check the status of your gateways, go to the "Devices" menu. Select "LoRa" in the connectivity selector and click on "LoRa gateways"
You have access to the list of your gateways

To see the details of a gateway select it in the list
2.4.8. MQTT devices

2.4.8.1. Add a MQTT device

Adding an MQTT device to Live Objects is automatic. When connecting to Live Objects for the first time, the device registers itself and uploads its own informations (firmware version, parameters). For this the equipment must have been properly configured. (See the developer guide https://liveobjects.orange-business.com/doc/html/lo_manual_v2.html#MQTT_MODE_DEVICE)

A number of attributes can be added to a device to facilitate its management in Live Objects (groups, properties, tags). These attributes are not stored at the device level but remain at the platform level.

It is also possible to add an MQTT device to Live Objects before it first connects to the platform. This allows for example to prepare a fleet by setting in advance the attributes above. For that just click on the button “Add an equipment” from the main menu “Devices”

![MQTT device configuration form](image)

**Name** : Allow to give a familiar name to the device

**MQTT Client Id** : This is the MQTT connection parameter of the same name used by the device to connect

**Device ID** : This is the unique identifier of the device in Live Objects. In the URN form (Unique Resource Naming). The namespace* as well as the identifier of the device inside this namespace can be customized.

**Default Stream Id** : This is the default time-serie in which messages issued by the device will be stored.
Group: Select a group in which to add the device. By default the device is added to the root group. (See paragraph « Groups »)

Tags: Free labels that can be used to ease sorting and identification of devices in the fleet

Properties: Free key-value couples that can be added to devices to enrich device management

* The namespace provides a consistent naming for a set of devices. For example all devices of the same manufacturer or same type. The id identifies the device inside this namespace.

2.4.8.2. Delete an MQTT device

It is possible to delete an MQTT device from the platform. This action will remove it from the Live Objects device manager (history, properties, groups etc.). Data issued by the device are not deleted and remain accessible in the "Data" menu.

To delete one or more devices, select the devices concerned by ticking the corresponding box in the list, and click on the "Delete" icon.

2.4.8.3. Details of an MQTT device

To access detailed informations about an MQTT device, select it in the list of devices in the main “Devices” menu.
The menu **Supervision** displays the status of the device and its connectivity interfaces.

- Date of last communication
- Alarm in progress, if an alarm rule has been defined for this device (See paragraph "Alarms")
- Status of the connectivity interface:
  - Connectivity status (See paragraph "Devices status")
  - MQTT connection settings

The menu **Identity** displays and modifies information about the device and its connectivity interfaces.
2.4.8.4. Update parameters of an MQTT device

A device can have specific parameters as factory parameters, or applications embedded directly in its micro code. These settings can be managed in Live Objects if the device support it. The device must have previously informed Live Objects about these parameters (publish to dev/cfg. See developer guide for details).

The visualization and the update of the parameters is possible in the menu "Parameters"
Informations displayed are:

- the name of the parameter (Id)
- its current value and its type
- its last update
- its status:
  - « new » means that the parameter has been reported by the device for the first time and has never been synchronized with Live Objects
  - « pending » means that an update request is pending to be delivered to the device, and will be delivered when the device will be ready to receive it (by subscribing to topic dev/cfg/upd)
  - « sent » means that the update request has been received by the device (subscribe to dev/cfg/upd), but the device didn’t confirmed yet the update (publish to dev/cfg with the new value of parameter). See developer guide for details
  - « sync » means that the parameter has been successfully updated by Live Objects. The value of the parameter announced by the device on topic dev/cfg is the same as the target set in Live Objects. The device and Live Objects have the same value of parameter.
  - “failed” means that the update request has been received by the device, but the device has answered to Live Objects with a different value of the parameters than those set as the target. See developer guide for details

To send an update request, click on a parameter and set its target value.
Click on button « Update » to validate the request, and repeat the operation for each parameter you want to change.

Changes to apply are listed in column « target value » with mention “change to send”. At this moment the requests are not yet sent. To send it to the device, you must click on the button « Send changes »
2.4.8.5. Send a command to an MQTT device

To send a command to an MQTT device, go to « Commands »

The history of commands is displayed. A command can have several statuses:

- **pending**: The command has been registered into Live Objects but is pending to be delivered to the device. For that the device must be connected and subscribe to topic dev/cmd
- **sent**: The command has been received by the device (subscription to dev/cmd), but the device didn’t acknowledge it yet (by publishing to dev/cmd/res). Note that if no applicative acknowledgment was requested when sending the command, the command automatically changes to the state "processed"
- **canceled**: The command has been canceled from Live Objects before having been received by the device
- **processed**: The command was sent successfully. The device received the command and Live Objects received the response from the equipment (if an acknowledgment was requested when the command was sent)
- **replied**: In this case where an application acknowledgment has been requested, this status indicates that the device responded by publishing on the dev/cmd/res topic.
- **expired**: The command couldn’t be delivered within the expiration delay set when sending the command
The MQTT topics to which the device must subscribe or receive the commands and on which it must publish the answers are described in the developer documentation: https://liveobjects.orange-business.com/doc/html/lo_manual_v2.html#_summary

To send a new command click on the button « Add command »

The command to send must be typed in the field "Request". It is possible to pass arguments with a command. These arguments must be entered as key-value pairs. NB: The values of type string must be enclosed in quotation marks.

You can select an expiration time for your command. After this delay, if the command could not be transmitted to the device (the device did not subscribe to the reception topic dev/cmd) then it goes into the "expired" state and will not be retransmitted.
Acknowledgment levels:

- **None**: Live Objects just ensures that the command has been sent to the device (subscription of the device to the reception topic dev/cmd and acknowledgment of the message at the MQTT protocol level)

- **Applicative**: Live Objects is waiting for a response message from the device (publication on the topic dev/cmd/res). The response message contains more information about the result of the processing of the command by the device. Though this processing can be an error from an application point of view, the command will appear as successfully processed (see dev guide for detail on configuration on device side [https://liveobjects.orange-business.com/doc/html/lo_manual_v2.html#_summary](https://liveobjects.orange-business.com/doc/html/lo_manual_v2.html#_summary))

You can configure the maximum time within which the device must give its acknowledgment as well as the number of new attempts to be made automatically in case of non-reception of the acknowledgment within this time.

Click on button «Validate» to send the command.

Note: as long as the device has not received the command (status pending), it is possible to cancel it, by clicking on the icon «Cancel»

This command can be sent back later, like any other command in the history by selecting it and clicking on the icon "Repeat".

Finally commands can be deleted. To do this select commands to delete and click on the icon «Delete».

**NB**: Deleting a command that has been sent ("sent" status) does not send a cancellation order to the device. It removes the command from the history and from the Live Objects device manager.

2.4.8.6. **Update the firmware of an MQTT device**

To update the firmware of an MQTT device go to the device details and select "Firmware".

The update screen is divided into two parts. The first part presents the firmwares that the device has announced and if update operations are in progress. The second part is the history of updates that have been made.
Note: Live Objects can manage several firmwares for a same device, if this device has several internal components (modem, core chipset etc) with each their own firmwares.

**Prerequisites**: To be able to update the firmware of your device,:

- The device must support this operation,
- This device has previously announced to the platform the name of its firmwares and its current versions (publish to topic dev/rsc). See developer guide.
- The firmware versions to manage have been previously created in the Live Objects Firmware Manager and the corresponding binaries imported into the platform. (See the "Configuration" menu for firmware management)

To start the update, select the firmware and click on the "Update" link. In the pop-up window, choose the target version and click on the "Save" button.

The status of the request changes to "pending". This means that the request is waiting for reception by the device (the device must subscribe to topic dev/rsc/upd to receive the request).
Once received by the device the request goes to "preparing_asset - 0%". This means that the device has received the request but has not yet responded if it accepts it. (To accept the request the device must answer by publishing on topic dev/rsc/upd/res)

If the device accepts the request, it turns to « waiting_transfer_info ». Once the device starts the download of the firmware, a percentage will show the progress of the transfer

Once the transfer is made, and the device has installed the firmware, the device goes announces to Live Objects its new version (by publishing on topic dev/rsc). If the version announced is the same as the target value that has been set, the process ends successfully and the status changes to "done". The history is also updated.
2.4.9. **SMS devices**

You can connect to Live Objects devices that communicate with SMS. These devices must have Orange SIM cards.

To manage SMS devices go to menu "Device" and select "SMS" from the drop-down list.
2.4.9.1. Add an SMS device

To add a new SMS device, click on the "Add a device" button in the "SMS" main menu.
**Server phone number:** This is the short number that will be used by your SMS device to communicate with the platform. This number was provisioned when you created your Live Objects account.

**Decoder:** Select decoder in the list if the payload needs to be decoded.

**MSISDN:** This is the number of your device in the format shown in the example.

**Device ID:** This is the unique identifier of the device in Live Objects in the URN form (Unique Resource Naming). The namespace (or "namespace ") as well as the identifier of the device inside this namespace ("id") can be customized.

**Default Stream Id:** This is the default time series in which messages issued by the device will be stored.

**Group:** The device group to add the object to. By default only the root group is offered. (See the "Groups" section)

**Tags:** Free typing label. Tags can be used to easily sort the devices (eg "demo"). You can enter up to 10 tags of 32 characters per device.

**Properties:** key-value pairs in free input. Example: device type, manufacturer, model ...

* The namespace provides a consistent name for a set of devices. For example, all devices of the same vendor or type. The id identifies the device in this namespace.

**2.4.9.2. Delete an SMS device**

To delete an SMS device, simply select it in the list and click on the "Delete" icon.

**2.4.9.3. Send a command to an SMS device**

To send a command to a SMS device go to the device details and click on the "Commands (Downlink)".
The history of commands sent to the device is displayed in a chronological order. The "status" column displays the status of the command and can have the following values:
  - **Pending**: Means that the command is pending in Live Objects. It has not yet been transmitted to the network (For example because other previous orders are still being processed)
  - **Sent**: The network sent the command to the device

The history of commands is exportable in a csv file.

To display the details of a command click on it.

A command can be sent again without having to reenter it, for that, you just have to select it in the list and to click on the icon "Repeat"

To send a new command, click the button "Add a command"

The data to be sent must be entered in the "payload" field in text format or in hexadecimal for binary format.
It is not currently possible to define the maximum duration during which the command can remain awaiting sending in Live Objects (status "pending") before being automatically canceled ("expiration" field present for LoRa and MQTT devices).

2.4.9.4. Modify an SMS device

To modify an SMS device, display its details and go to the "identity" menu. You can modify the information specific to a device (group membership, tags, properties), or the characteristics of its connectivity interface (Server number or decoder assigned). To do this click on the "modify" icon.

![Modify an SMS device](image)

2.4.10. External connector

The external connector makes it possible to manage devices in Live Objects completely agnostically from connectivity. Thanks to the external connector, devices managed by a proprietary cloud, or using specific communication protocols, can benefit of the same device management features of Live Objects (eg campaign management, groups, etc.) as other MQTT, LoRa or SMS devices. Device management is done via unified interfaces and APIs.


The management of devices with external connector is similar to that of MQTT devices. To provision a device of "external connector" type, there are two possibilities, as for MQTT:
1 / Either by autoprovisioning: The device is automatically created in the Device Manager of Live Objects during the first connection (see documentation indicated above)
2 / Either by creating the device in the Device Manager before its first connection, thanks to the Live Objects portal. To do this, go to the menu “Devices” and click on “Add new device”

The "external connector ID" field is the id of the device in the external connector. This id is used to build the device id in Live Object.

The other fields, "streamId", "tags", "groups" and "properties" are the same as for other types of devices (see the corresponding paragraphs)

The management of devices with "external connector" is very similar to that of MQTT ones. In particular, the command delivery, which has the same characteristics and statuses, with the difference that in connector mode the states displayed in Live Objects are those reported by the connector and not by the end device itself.

Currently, the external connector allows data report, sending command with or without acknowledgment, as well as sending mass commands (see campaign management). Management of firmware and parameters is not currently available in external connector mode.

2.4.11. Groups

You can organize your fleet with groups of devices.
You can create, rename, or delete a group, or a subgroup by clicking at the root "All Devices" or an existing group or subgroup.

Note: It is not possible to delete a group that contains at least one device.

By clicking on a group you will see the list of devices that are part of it. A device can only belong to one group at a time.

To move a device from one group to another, go to the device details, or select it from the list and click on the "Move" icon.
2.4.12. Campaign management

Campaigns enable to plan, execute, and track mass device management operations. Allowed operations are those supported by the connectivity used by the devices.

2.4.12.1. Create a campaign

To manage campaign go to menu « Devices », and click on « Campaigns »,

then click on « Create a campaign »

The first step is to select the targets. This can be done:

- Either by importing a CSV file: Either by importing a list of existing devices retrieved from a csv export of the fleet (see the paragraph “Export a list of devices”), or by importing a list of devices that does not yet exist in Live Objects; in this case the import file will contain only the column "Id".
Either by using advanced filters

The second step is to indicate the type of operation that will be applied to the selected devices, and to provide the parameters corresponding to the operation. Allowed operations are those supported by the connectivity used by the devices (Refer to the paragraph corresponding to your connectivity in this document)
NB: The Live Objects portal allows you to choose only one operation to apply to target devices. If you need to combine within the same campaign several operations per device (e.g. a command, followed by a firmware update) you can do it using the APIs https://liveobjects.orange-business.com/doc/html/lo_manual_v2.html#_campaign_management and https://liveobjects.orange-business.com/swagger-ui/index.html#!/Campaign_management

NB: For each operation it is possible to specify a number of retries in case of failure (up to 5)

The third step consists in plan the campaign.
2.4.12.2. Follow a campaign

Campaigns are sorted depending on their planning: «Scheduled», «Ongoing» et «Completed». Click on the corresponding tab to display them.
2.4.12.3. Cancel a campaign

It is possible to cancel a campaign by clicking on the "Cancel" icon.

If the campaign is already running, and the option "Force cancellation of the campaign" is not checked, the status of the campaign changes to "Cancel" until the current actions are completed. Actions not started do not start.

If the option "Force cancellation of the campaign" is checked, the current actions are canceled and the campaign is immediately stopped.

Canceled campaign remains in campaign history with status "Canceled"

2.4.12.4. Delete a campaign

It is possible to delete a campaign by clicking on the "Delete" icon.

If the campaign is running or canceling, it cannot be deleted, except by selecting "Force campaign deletion".

In this case all current actions are canceled. And the campaign is removed from the campaign history.
2.4.12.5. **Status of a campaign**

<table>
<thead>
<tr>
<th>Status</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scheduled</td>
<td>the start date is not yet reached</td>
</tr>
<tr>
<td>Ongoing</td>
<td>• the start date is reached</td>
</tr>
<tr>
<td></td>
<td>AND</td>
</tr>
<tr>
<td></td>
<td>• the end date is not yet reached <strong>and</strong> at least one action is still ongoing on a target</td>
</tr>
<tr>
<td></td>
<td>When a campaign is running, its progress is displayed: number of targets reached / total number of targets, as well as failed targets</td>
</tr>
<tr>
<td>Canceling</td>
<td>the campaign is being canceled. Waiting for the current actions to finish. Actions that did not start will not start</td>
</tr>
<tr>
<td>Canceled</td>
<td>the campaign has been canceled. Some scheduled actions may not have been completed</td>
</tr>
<tr>
<td>Success</td>
<td>the campaign ended without error: All targets were reached and the operations successfully completed*</td>
</tr>
<tr>
<td></td>
<td>(* See corresponding paragraphs for the actions concerned)</td>
</tr>
<tr>
<td>Incomplete</td>
<td>the end date has expired, but targets could not be reached, or some operations failed.</td>
</tr>
</tbody>
</table>

2.4.12.6. **Status of a target device**

<table>
<thead>
<tr>
<th>Status (EN)</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Success</td>
<td>the operations were carried out successfully* on the target device (* See the corresponding paragraphs for the concerned actions)</td>
</tr>
<tr>
<td>Not started</td>
<td>no operation has been started yet on the target device</td>
</tr>
<tr>
<td>In progress</td>
<td>an operation is ongoing on the target device</td>
</tr>
<tr>
<td>Failure</td>
<td>an operation has failed on the target device</td>
</tr>
</tbody>
</table>
2.4.12.7. Campaign report

The detailed status of a campaign can be consulted at any moment. For that just click on it on the list.
2.5. Data

Data stored in Live Objects can be visualized in the menu « Data »

The data display can be filtered using different fields:
- **From / To**: The time range to search
- **Connectivity**: This is the type of connectivity that your device uses to communicate with the platform. It can be "loras", "mqtt", "sms"
- **Source**: The source that have produced the data (e.g. URN of a device)
- **Stream**: This is the time series in which the data are stored. A stream can contain all the data sent by the same device (default for LoRa), or all data of a given type (e.g. "temperatures", "pressure" …)
- **Tags**: Labels inserted in the data message by the device that issued it. You can enter up to 10 tags of 32 characters per device

For each data displayed in the table:
- **Date**: This is the date of the data communicated by the device (or the gateway in the case of Lora), thanks to the "timestamp" field of the data message. (NB: If this field not provided, the date is the date of arrival of the data in the platform)
- **Tags**: Like devices, data can be tagged. This tag is inserted in the data message by the device that emits it.
- **Value**: This is the "data message" produced by the device. Details are available by clicking on it.
- **Other details specific to the type of device (LoRa or MQTT connectivity)
### Uplink message details

<table>
<thead>
<tr>
<th><strong>Message Information</strong></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Payload</strong></td>
<td>417c75af500000000</td>
</tr>
<tr>
<td><strong>Fcst</strong></td>
<td>5</td>
</tr>
<tr>
<td><strong>Port</strong></td>
<td>5</td>
</tr>
<tr>
<td><strong>Decoder</strong></td>
<td>demo</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Connectivity information</strong></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Network signal</strong></td>
<td>null</td>
</tr>
<tr>
<td><strong>RSSI</strong></td>
<td>-84</td>
</tr>
<tr>
<td><strong>Snr</strong></td>
<td>8.25</td>
</tr>
<tr>
<td><strong>Sf</strong></td>
<td>11</td>
</tr>
<tr>
<td><strong>Gateway count</strong></td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Customer tags</strong></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>demo</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Location</strong></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Lat/Lon</strong></td>
<td>48.600419,2.295627</td>
</tr>
<tr>
<td><strong>Type</strong></td>
<td>lora</td>
</tr>
</tbody>
</table>
Decoded data

```json
{
  "npush": 0,
  "temperature": 26.01913
}
```

Details

```json
{
  "metadata": {
    "source": "urn:lor:lora:0004A100801EF04A",
    "connector": "lora",
    "encoding": "demo",
    "network": {
      "lora": {
        "signalLevel": 5,
        "rsli": -84,
        "gatewayCnt": 3,
        "sfl": 11,
        "port": 5,
        "snr": 8.15,
        "location": {
          "slt": 0,
          "accuracy": 1500,
          "lon": 2.2956,
          "lat": 41.680419
        },
        "cnt": 6,
        "devEUI": "0004A100801EF04A"
      }
    },
    "streamId": "urn:lor:lora:0004A100801EF04A",
    "created": "2018-07-16T12:03:15.666Z",
    "extra": null,
    "location": {
      "provider": "lora",
      "slt": 0,
      "accuracy": 1500,
      "lon": 2.2956,
      "lat": 41.680419
    },
    "model": "lor_v6",
    "id": "5559352570c56514f172db",
    "value": {
      "npush": 0,
      "temperature": 36.01913,
      "payload": "41d76f5f00000000"
    },
    "timestamp": "2018-07-30T12:05:15.311Z",
    "tag": ["demo"]
  }
}
```

2.5.1. Data export

Filtered data can be exported to a csv file. Just click on the “Export” icon

2.5.2. Advanced visualization (Kibana)

Live Objects relies on "Kibana" an open source tool for data mining and dashboarding. It is based on "Elastic Search" technology that allows dynamic dashboards and advanced data search. Go to https://liveobjects.orange-business.com/#/cms/ressources-dashboards to learn more about Kibana and other tools usable with Live Objects

To launch Kibana click on the button "Kibana"
Prerequisites for MQTT devices to have data been indexed by Elastic Search:

- the "model" field must be entered in the data message returned by the device. This field provides information on the structure field “value” of the data message and allows the indexing of the data contained in this field.
- if the field “model” is not present in the data message, the data in "value" will not be indexed
- if the structure of the field “value” changes (for example a "temperature" field that goes from integer to string), the value of the field "model" must also be changed. The same data model cannot be used to index data of different structures.
- the value of the field "model" is not imposed by Live Objects, it is provided by the device

See developer guide for detailed description of a data message:
For Lora devices

- the payload must be decoded so that the data can be indexed
- the decoder must also specify a field “model” corresponding to the structure of the decoded data, for the same reasons as explained above

2.5.3. Messages queue (fifo)

Message queues, or "fifo" are limited disk space used to persist messages consumed in real time by business applications connected to Live Objects in MQTT (see paragraph "Data consumption by applications").

The total disk space allocated to fifo depends on your offer (eg 100 MB for 10 fifo). Contact your customer service for more information.

The messages are stripped of the fifo (first entry, first out) as soon as one or more consumers have subscribed. In case of several consumers on the same fifo messages are "load balanced" between consumers.

If a fifo is full the most recent message replaces the oldest message in the fifo.

Messages not consumed in fifos are automatically deleted after 7 days.

To create a fifo please go to the "Data" menu and select the fifo section.

To feed a fifo with data see the paragraph "Routing data".
2.5.4. Routing data and events

Data collected by Live Objects from devices can be routed to message queues (fifo) or pushed to external http servers (push http). The devices state changes* (connection, disconnection, activation, reactivation, etc.) can be routed as well to your application servers. (* NB See “Device status” paragraph)

2.5.4.1. Create a routing rule

Creating a routing rule is done in three steps:
1) Name of the rule
2) Criteria for selecting messages to be routed
3) Mode of routing (fifo or push http)

To get started go to the "Routing" section of the "Data" menu, and click "Add a routing rule"
Step 1: Naming of the rule

You must first choose what type of message you want to route: data sent by your sensors ("Data message"), or device management events, in this case devices state changes (for example for supervision purpose or implementation of business scenarios).

Then, if you want to route only certain messages, you can filter them:
- By connectivity: Only messages from devices that use the selected connectivity will be taken into account
- By groups or subgroups: Only messages sent by the devices belonging to the selected groups will be taken into account.
• You can also directly enter the device identifiers (URNs) whose messages you want to route. You can get device IDs by viewing their details in the "Devices" menu. (Available only for data routing)

NB: Criteria by "group" or by "equipment identifier" are mutually exclusive

Step 3: Select routing mode

See the following paragraphs to set the routing mode. NB: It is quite possible to combine several routing modes (eg: routing to both fifos and external http servers)

2.5.4.2. Routing to fifo

At step 3 of routing rule creation, click on the "+ FIFO" button, then select the target fifo from the list of existing fifos.

To create a fifo go to the paragraph "Messages queues (fifo)"
You can route messages to multiple fifo at a time. In this case the messages will be duplicated in each of target fifo

### 2.5.4.3. Push http

At step 3 of routing rule creation, click the "+ HTTP push" button.

In the URL field, enter the address of the external http server to send the data to.

If the external http server requires specific headers (typically authentication information) you can enter them in the corresponding fields.

**Warning:** allowed outgoing ports are standards ones only: 80, 8080, 443, 8443, 9243

If the "Failure policy" is enabled, Live Objects will try, in case of a "5xx" errors returned by the remote server, to transmit the messages again at regular intervals.

NB: Only the errors indicating a malfunction of the remote server (ex: 500 Internal error) are taken into account by the retransmission mechanism. The type 400 errors (page not found, access forbidden ...) do not indicate a priori abnormal behavior of the remote server, the retransmission mechanism will not be activated in these cases.

The data to be transmitted can be:

- **The whole message:** This is the entire Live Objects message, ie the data sent by the equipment itself ("value" field: {...}) as well as all metadata (eg connectivity information ...) and data from Live Objects Device Enrichment (group membership, properties etc)
- **Value only:** Only the "value" part: {...} of the Live Objects message

![HTTP Push Configuration](image-url)
• **A Mustache formatted message:** In this case you can choose which data contained in Live Objects messages you want to transmit and in which form. The formalism used for formatting is the "Mustache" formalism [https://mustache.github.io/mustache.5.html](https://mustache.github.io/mustache.5.html)
Example below:

```json
{
  "metadata": {
    "topic": "event",
    "source": "urn:.orange:android:35709970479549PRIMARY"
  },
  "stream": "android35709970479549PRIMARY",
  "created": "2018-11-08T4122:30.157Z",
  "extra": null,
  "location": {
    "provider": null,
    "alt": null,
    "accuracy": null,
    "lon": 2.4695,
    "lat": 49.0378
  },
  "model": "deep",
  "id": "5be44e8e74a91bb3e9b5fd4",
  "value": {
    "rev": 445,
    "co2": 445,
    "doorOpen": false
  },
  "hygrometry": 62,
  "temperature": 11,
  "pressure": 891
}
```

Important: You have the possibility to test your webhook before creating your routing rule to ensure that Live Objects communicates properly with the external service. For this, use the test API available in the swagger [https://liveobjects.orange-business.com/swagger-ui/index.html#!/Triggers_and_Actions_-Test/testHttpPushUsingPOST](https://liveobjects.orange-business.com/swagger-ui/index.html#!/Triggers_and_Actions_-Test/testHttpPushUsingPOST)

2.5.4.4. **Suspend/Enable a rule**

You can at any time suspend or reactivate a rule via the corresponding icons in the "Routing" section of the "Data" menu.

2.5.5. **Data consumption by applications**

Business applications have different APIs for consuming Live Objects data, in real time or asynchronously.
2.5.5.1. Explore data

For data exploration, the REST http APIs should be used:

- **Search API:** [https://liveobjects.orange-business.com/swagger-ui/index.html#!/Data_management_data_search/dslQueryUsingPOST](https://liveobjects.orange-business.com/swagger-ui/index.html#!/Data_management_data_search/dslQueryUsingPOST). This API is suitable for complex searches. The searched data must be indexed.

For more details on these APIs please consult the developer guide: [https://liveobjects.orange-business.com/doc/html/lo_manual_v2.html#DATA](https://liveobjects.orange-business.com/doc/html/lo_manual_v2.html#DATA)

2.5.5.2. Consume data in real time

Data can be consumed in real time by business applications in two ways: Either by connecting to Live Objects with MQTT protocol, or by asking Live Objects to push the data to the application server via http.

2.5.5.2.1. Consume data with MQTT

**Prerequisites:**
- Have created a fifo and feed it data by a routing rule. See the paragraphs "Message Queues (FIFO)" and "Routing Data"
- Code samples are available to integrate the MQTT protocol into your application: See [https://liveobjects.orange-business.com/#/cms/ressources-exemples-de-codes/](https://liveobjects.orange-business.com/#/cms/ressources-exemples-de-codes/)

NB: The use of MQTT clients with GUI can greatly facilitate the testing and prototyping of MQTT connections. We can mention HiveMQ, MQTT.box, MQTT.fx.

**Step 1: Connection**

The MQTT connection parameters to be used are the following:

- **Server URL:** liveobjects.orange-business.com
- **Port:** 1883 (default), 8883 (MQTTS), 80 (MQTT over websocket), 443 (Secure websocket)
- **Username:** application
- **Password:** the value of an API key with BUS_R privileges (see "Manage API Keys")
- **Keep Alive:** 30 seconds

NB: The MQTT connection identifier ("clientID") is free.

NB: The MQTT "will" options are not used by the MQTT Live Objects server

**Step 2: Subscription to the fifo**

Subscribe to the fifo that was previously created (see prerequisites) by subscribing to the topic: fifo / myFifo

The prefix fifo / is mandatory

myFifo is the name of your fifo

The consumption of the messages starts as soon as the subscription is carried out.

NB: The values of "qos" MQTT supported by Live Objects are 0 and 1.

**2.5.5.2.2. Consume data with push HTTP**

**Prerequisites:** An application server with a web service able to receive HTTP requests from Live Objects

To start message consumption via http push, create a "push http" routing rule. Messages are pushed to the target application server as soon as the rule is activated. See the paragraph "Routing data / Push http"

**2.5.6. Manage decoders**

A decoder allows translating data issued by a device from a "computer" form (e.g. hexadecimal payload of a LoRa device) into a human understandable and indexable data. This translation also allows usage of advanced services based on data, such as event processing. These advanced features are currently only available with API (See developer manual [https://liveobjects.orange-business.com/doc/html/lo_manual_v2.html](https://liveobjects.orange-business.com/doc/html/lo_manual_v2.html))

**2.5.6.1. Type of decoders**

Decoders available in Live Objects are of 2 types:

- "Configurable" decoders: These are binary decoders (for hexadecimal payloads), or csv (text payload with separator). These decoders are suitable for devices whose payload structure does not vary. You can create your own configurable decoders yourself (see paragraphs after)
- "Programmable" decoders (Script): These decoders are intended for devices whose payload structure are complex and cannot be processed by a configurable decoder. For security reasons, the deployment of this type of decoder can only be done by Orange teams (see paragraph "Create a programmable decoder")
2.5.6.2. Visibility of decoders

Decoders can be "private", that is to say visible and usable only in your Live Objects tenant; this is the case of the configurable decoders that you create yourself.

Orange also offers as part of its catalog of objects https://iotmarket.orange.com a list of predefined decoders adapted to different hardware. These decoders are called "public" that is to say, visible and usable by all Live Objects customer.

The list of public and private decoders is displayed in the menu "Decoders".

![Decoders list](image)

2.5.6.3. Data model and indexing

When you create a decoder you can specify a model. This model describes the structure of the decoded data and allows its correct indexing in Live Objects. If no template is specified, a default template will be assigned:
- "lora_v0" in the case of data issued by LoRa equipment
- "sms_v0" in the case of data sent by SMS equipment
- There is no default model in the case of an MQTT device which means that the decoded data will not be indexed if no model is explicitly indicated

Example, let’s take the decoded data below:

```json
"value": {
    "test3": 0,
    "test5": 7,
    "pressure": -86,
    "temperature": -1
}
```

If the model defined when creating the decoder is "MyModel", then the decoded data will be indexed in Live Objects as follows:

@mymodel.value.temperature,
@mymodel.value.pressure etc

If no model is specified and the data comes from a lora device, the decoded data will be indexed as follows:

@lora_v0.value.temperature, @lora_v0.value.pressure etc

**IMPORTANT**: A same data cannot be indexed with two different types. Example, here the information carried by the variable "temperature" becomes a string and no longer numeric

```json
"value": {
    "test3": 0,
    "test5": 7,
    "pressure": -86,
    "temperature": "-1"
}
```

If the model used is the same as before then there will be an indexing conflict, and the data @mymodel.value.temperature will not be indexed. In this case an error will be reported in the activity log (see section "Activity Log").
2.5.6.4. Create a binary decoder (configurable)

To create your own configurable decoder, click on the button « Add ».

Decoder creation is in two steps:
1. Definition and test
2. Creation

**Step 1: Definition of the decoder**

In this step you have to describe the structure of the payload. Live Objects relies on the Java Binary Block Parser (JBBP) for the description of binary structures. The different types are available here [https://github.com/raydac/java-binary-block-parser#primitive-types](https://github.com/raydac/java-binary-block-parser#primitive-types)
In the field «Binary payload description» enter the structure of your payload as follows: variable type;

In the example given below, the variable "test3" is coded on 3 bits, the variable "test5" on 5 bits and the variables "pressure" and "temperature" on 1 byte each. This constitutes a payload of 3 bytes in total.
It is recommended to test the decoder before creating it. To do this, do not immediately click on the "Create" button, but first enter a test payload in the field "Payload to be decoded, in hexadecimal format" and click on the "Test" button. In the above example the 38AAFF payload will match the decoded values

```json
{
    "test3": 0,
    "test5": 7,
    "pressure": -86,
    "temperature": -1
}
```

Indeed 38AAFF matches to the binary sequence 00111000 10101010 11111111

First byte: the three first bits are 000 that equals to 0 in decimal (value of "test3"), the following five are 00111 that equals to 7 in decimal (value of "test5"), so the entire byte 00111000 equals to **38 in hexadecimal**

Second byte: AA in hexa equals -86 in decimal (value of "pressure")

Third byte: FF in hexa equals -1 in decimal (value of "temperature")
It is possible to format the decoded data and to apply arithmetical operations (offset application, unit conversion ...). This is possible through the use of templates. The templates are based on the "mustache" formalism (see http://mustache.github.io/)

If we take the previous example we may want to give a particular meaning to the values represented by "test3" and "test5" and convert the pressure and temperature values and display them with the correct unit.

The selector "Template mathematical evaluation" allows to restrict the mathematical evaluation to the tags {{math}} only (value off), or to extend it to the whole template (value on)

**Warning**: The character "." in variable name is not supported. In the previous example "bat.stat" instead of "batStat" would cause an error.

**Step 2: Creation of the decoder**
Once the decoder has been validated, you can create it by clicking on the "Create" button. It becomes immediately available for use by your devices (see the paragraph "Assign a decoder to a device")
2.5.6.5. Create a CSV decoder (configurable)

To create a csv decoder, proceed in the same way as for a binary one. The structure of the payload is simply composed of "Name of the variable" (name) and "type" (jsonType), and a separator character. It must be entered in the "Columns descriptions" field.

Possible types are: NUMERIC (integer, long, float double), STRING (UTF-8), and BOOLEAN ("true"/ "false")

The field « Parser options » allows specify different options of csv format. If left blank the default separator is the comma.

See the developer guide for the complete list of available formatting options https://liveobjects.orange-business.com/doc/html/lo_manual_v2.html#_csv_decoding

It is also possible to define a template to format decoded data, in the same way as for the binary decoder (see *Create a binary decoder*)
2.5.6.6. Create a programmable decoder

If your device produces payloads with variable and complex structures, you can use programmable decoders. The creation of this type of decoder requires intervention on the part of Orange teams. If you cannot find a public decoder corresponding to your device, please make your request at liveobjects.decoder@orange.com.

Two options are available to you:
1/ you want to make your decoder yourself. For this you must use the framework available here [https://github.com/DatavenueLiveObjects/Payload-decoders](https://github.com/DatavenueLiveObjects/Payload-decoders) and follow the guidelines. Once your decoder has been developed and tested, you will communicate it to Orange teams, who will make a technical acceptance and put it into production.

Or

2/ you ask Orange to develop the decoder for you. For this you must communicate to Orange teams the detailed specifications of the payloads emitted by your device as well as test data for each decoding case (binary payload, and decoded payload)

2.5.6.7. Disable/enable a decoder

When it is created, a decoder is automatically activated, but it is possible to deactivate it for temporarily suspend the decoding for all devices that uses it.

To disable/enable a decoder, select it in the main list, then click on the "Disable / Enable" button.
2.5.6.8. Assign a decoder to a device

You can assign decoders to LoRa or SMS devices when they are created, or after by going to the device details, “identity” menu. (See the paragraph corresponding to your type of device).

NB: Once the decoder associated with the device, the next payloads will be automatically decoded. The payloads issued before the association of the decoder remain not decoded.

The way of decoders work with MQTT devices is different. In this case there is no prior assignment of a decoder to a given device, it is the device itself which chooses the decoder which it uses at the time of the publication of its data via a topic specific to binary mode (see https://liveobjects.orange-business.com/doc/html/lo_manual_v2.html#MQTT_DEV_ENCODED_DATA).

2.5.7. Activity Logs

The activity log is a valuable tool for diagnosing connectivity, storage, or notification issues to third-party systems.

The information displayed in the activity log can be filtered by category, and offers enabled on the Live Objects account.

To access the activity log, go to the Data menu and click on “Activity logs"
2.5.7.1. LoRa connectivity

To view the LoRa connectivity information, select in the filter bar, the category "Connectivity" and the value "LoRa".

The logs you can have for LoRa connectivity are:

- Join Request, join Accept
- MAC commands with display of raw values for commands initiated by the device (LinkCheckReq/LinkCheckAns and DevStatusReq/DevStatusAns)
- Uplink: applicative payload sent by the device, Request for network acknowledgement (ack Uplink)
- Downlink: Command sent from Live Objects to the device, uplink acknowledgement

2.5.7.2. MQTT connectivity

To view information about MQTT connectivity select in the filter bar, the category "Connectivity" and the value "MQTT".

The following events are logged with ERROR level:

- Subscription / publication to / on an incorrect topic
- Publication with an incorrect message format
- Rate limiting achieved
- Logout due to a concurrent connection using the same clientId
The following events are logged in INFO level
- Connection / disconnection

NB: The INFO level is not logged by default and requires to put the Key API in debug mode. To enable debug mode on an API key refer to the section *Put an API Key in Debug Mode*

(1) Consult the developer guide
(2) Consult your customer service

2.5.7.3. Storage

The storage category is used to log events related to indexing issues:
- Limit of number of indexable fields reached (1,000)
- Pattern inconsistency (One field can not be indexed with two different types for the same model)
- Data impossible to index (Ex: An array containing data of different types [1234, "data"])

2.5.7.4. Notification

The notification category is used to log notification events to external systems:
- Push http: Only errors are logged. Success pushes (http 20x return codes) are not logged.

2.5.7.5. Limitations

- Events are kept in the audit log for 14 days
- The maximum number of events logged is 10 million
- The maximum number of events that can be exported via the portal is 10,000

2.6. Configuration

The menu « Configuration » allows the management of your Live Objects account.

2.6.1. Identification of the account

Here are the informations that identify your Live Objects account:
- **Name**: The name of the Live Objects tenant
- **Client account number**: Your Orange customer ID as it is in your bill
- **Live Objects ID**: It’s a technical id, useful for technical support
- **Creation date**: Creation date of the Live Objects tenant
2.6.2. Identification of the current user

Several users can connect to the Live Objects account (administrators, simple users. See paragraph “Manage users”). To know with which user you are connected, go “My profile”.

2.6.3. Traffic limits

The traffic limitations applied to your Live Objects account depend on your offer. To know them go to the section “traffic limits” in the menu Configuration>Account
2.6.4. Roles and rights in Live Objects

Rights in Live Objects are managed through "roles". Each role defines a scope of possible actions on one or more features of the product. These roles can be assigned to users, or API keys used by devices or applications to access to Live Objects.

Two user profiles are available in Live Objects:
- « administrator » : it has all administrator rights on the Live Objects account,
- « user» : it has mainly read rights

It is also possible to define a custom profile, by selecting each role manually. (See paragraph « Manage users»)

The table below summarizes the roles attributed to each profile:

<table>
<thead>
<tr>
<th>Technical value (Read/Write)</th>
<th>Description</th>
<th>Administrator</th>
<th>User</th>
</tr>
</thead>
<tbody>
<tr>
<td>API_KEY_R/W</td>
<td>API keys management</td>
<td>R/W</td>
<td>R/W</td>
</tr>
<tr>
<td>USER_R/W</td>
<td>Live Objects users management</td>
<td>R/W</td>
<td></td>
</tr>
<tr>
<td>SETTINGS_R/W</td>
<td>Live Objects account configuration (for future use)</td>
<td>R/W</td>
<td>R/W</td>
</tr>
<tr>
<td>DEVICE_R/W</td>
<td>Device Management</td>
<td>R/W</td>
<td>R</td>
</tr>
<tr>
<td>DATA_R/W</td>
<td>Collected Data Access</td>
<td>R/W</td>
<td>R</td>
</tr>
<tr>
<td>DATA_PROCESSING_R/W</td>
<td>Data Decoders and Events Processing management</td>
<td>R/W</td>
<td>R</td>
</tr>
<tr>
<td>KIBANA_R</td>
<td>Access to Kibana</td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>BUS_CONFIG_R/W</td>
<td>Management of routing keys and message queues (fifo)</td>
<td>R/W</td>
<td>R</td>
</tr>
<tr>
<td>Access</td>
<td>Description</td>
<td>R/W</td>
<td>R/W</td>
</tr>
<tr>
<td>--------</td>
<td>------------------------------------------------------------------------------</td>
<td>-----</td>
<td>-----</td>
</tr>
<tr>
<td>BUS_R/W</td>
<td>Access to message bus access using MQTT or HTTP</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DEVICE_ACCESS</td>
<td>MQTT Device mode access (allow to create API keys for devices)</td>
<td>R/W</td>
<td>R/W</td>
</tr>
<tr>
<td>CAMPAIGN_R/W</td>
<td>Management of massive device management operations</td>
<td>R/W</td>
<td>R</td>
</tr>
<tr>
<td>LOGS_R</td>
<td>Allow access to the Activity Log</td>
<td>R</td>
<td>N/A</td>
</tr>
<tr>
<td>CONNECTOR_ACCESS</td>
<td>Allow to create API key to connect to MQTT external connectors</td>
<td>R/W</td>
<td>R/W</td>
</tr>
</tbody>
</table>
2.6.5. Manage users

To manage Live Objects users go to menu “Users”.

**Enabled**: Means the user has access to Live Objects

**Suspended**: Means the access to Live Objects for the user has been suspended by an admin user of the account

**Disabled**: Means that an activation email has been sent to the user, but the user didn’t activate it yet
2.6.5.1. Create a user

To create a new user click on the button “Add”

**Login** is used to log into Live Objects Portal.

**Email** is the contact email address where the activation email and password recovery emails will be sent.

**Profile** allows to select predefined right profiles, or to selects custom rights.

Click on button « Add » to validate the user creation. An activation email is immediately sent. This email contains an activation link with **7 days validity**.
As long as the user has not activated his access by clicking on the activation link, his status remains "inactive".

If the user has not validated his access within 7 days, it will be necessary to send again an activation email. This can be done by selecting the user(s) concerned and clicking on the "Send activation email" icon, or in the user's details by clicking on the "Send activation email" button.

2.6.5.2. Suspend / reactivate a user

It is possible to suspend access to the service for a user temporarily. For this go to the details of the user by clicking on it in the main list and click on the button "Suspend". The suspension is immediate, the user can no longer log in to the Live Objects portal.

To reactivate a suspended user, go to the user's details and click the "Enable" button. The reactivation is immediate and the user can connect again to the Live Objects portal.

2.6.5.3. Delete a user

To delete a user select it from the main list and click on the "Delete" icon. The deletion is immediate and definitive.

2.6.6. Manage API keys

API keys allow:
- to connect MQTT devices to the platform
- business applications to use Live Objects services through APIs (MQTT and REST):
  - Data mining (Search)
  - Real-time data consumption
  - Device management
  - Live Objects configuration (users, decoders etc.)

All features available through the Live Objects portal are also available through the REST APIs (see https://liveobjects.orange-business.com/#/swagger)

The MQTT APIs are reserved for:
- connecting MQTT devices ("device mode")
- real time data consumption by business applications ("bridge mode" with payload username)


As for users, it is possible to assign roles to API keys, to allow certain features or not. For example:
- an API key used by an MQTT device only requires the DEVICE_ACCESS role
- A business application that needs to use the device management APIs will require the role DEVICE_R (if simple consultation) and DEVICE_R + DEVICE_W (if consultation and modification).
- A business application that needs to consume real-time data with MQTT will need the BUS_R role.

NB: For each REST API, the necessary role is indicated in the swagger (https://liveobjects.orange-business.com/#/swagger).

To view all the API keys created in your Live Objects account, go to the "API Keys" section of the "Configuration" menu.

2.6.6.1. Create an API key

To create a new API key, click on the button "Add" in the menu "Api keys".
When creating an API key you can choose to assign to it a predefined rights profile based on your usage, or a custom profile.

- The "MQTT Equipment" profile is the profile to select for the keys used for connecting MQTT devices.
- The "Application" profile is the profile to be selected for the keys used by application servers on top of Live Objects for the usual use cases: real-time data consumption, configuration of message queues, management of decoders and "event processing".
- The "External Connector" profile is the profile to select for the keys to be used by external connectors. An external connector is used to manage in Live Objects "cloud" devices that cannot be connected directly to Live Objects (proprietary or unsupported connectivity, data model or operating mode)
- For more advanced needs ("Device Management", campaign management for example), use the "Custom" profile and select the appropriate rights.

You can also set a validity period for your key. At the end of the period the key will be automatically deactivated.

It is also possible to restrict an API key to one or more message queues (fifo). This means that this key can only be used for the consumption of routed messages into the specified queues and only that. The consumption of messages on other fifo, or the use of other APIs, including REST, with this key will be impossible.

This feature allows you to share data from some devices with a partner without giving access to other part of you Live Objects tenant.

Finally, you can associate a key with a certificate from your own certification authority, by selecting the "strong authentication" option. See the paragraph "Associate a certification authority"

To validate the creation of the key, click on the "Create" button. Your key is generated in the form of an alphanumeric sequence and a QR code.
Important note: The key is not stored in Live Objects. Only an hash is kept in the platform. It is your responsibility to keep it in a safe place. In case of loss the key cannot be found in Live Objects, it will have to be deleted or regenerated and the systems using it (MQTT device, applications) will have to be updated with the new key.

2.6.6.2. Hierarchy of API Keys

It is possible to create children of an API key. These children keys will automatically inherit roles from the parent key as well as its validity. It is not possible to add roles to a child that are not owned by the parent key.

Disabling the parent key automatically disables all of its children.

The use of child keys can be useful for managing groups and subgroups of API keys. For example, a single API key for a device group, with a daughter key for each, will disable access to Live Objects for that device group in one click.

To create a child key, simply select the key by clicking on it in the main list, then click on the "Add" button in the modification screen, at the bottom of the page.
<table>
<thead>
<tr>
<th>Roles</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User</td>
<td>Users management</td>
</tr>
<tr>
<td></td>
<td>Assign USER_R and USER_W roles</td>
</tr>
<tr>
<td>API Key</td>
<td>API Keys management</td>
</tr>
<tr>
<td></td>
<td>Assign API_KEY_R and API_KEY_W roles</td>
</tr>
<tr>
<td>Queues access</td>
<td>Message Bus access using MQTT or HTTP</td>
</tr>
<tr>
<td></td>
<td>Assign BUS_R and BUS_W roles</td>
</tr>
<tr>
<td>Queues config</td>
<td>Routing Keys and Message Queues management</td>
</tr>
<tr>
<td></td>
<td>Assign BUS_CONFIG_R and BUS_CONFIG_W roles</td>
</tr>
<tr>
<td>Campaign</td>
<td>Management of massive operations on a set of Devices</td>
</tr>
<tr>
<td></td>
<td>Assign CAMPAIGN_R and CAMPAIGN_W roles</td>
</tr>
<tr>
<td>Data</td>
<td>Collected Data Access</td>
</tr>
<tr>
<td></td>
<td>Assign DATA_R and DATA_W roles</td>
</tr>
<tr>
<td>Data Processing</td>
<td>Data Decoders and Events Processing management</td>
</tr>
<tr>
<td></td>
<td>Assign DATA_PROCESSING_R and DATA_PROCESSING_W roles</td>
</tr>
<tr>
<td>Device</td>
<td>Device management</td>
</tr>
<tr>
<td></td>
<td>Assign DEVICE_R and DEVICE_W roles</td>
</tr>
<tr>
<td>Device access</td>
<td>Device mode MQTT access</td>
</tr>
<tr>
<td></td>
<td>Assign DEVICE_ACCESS role</td>
</tr>
<tr>
<td>Account</td>
<td>Account configuration</td>
</tr>
<tr>
<td></td>
<td>Assign SETTINGS_R and SETTINGS_W roles</td>
</tr>
<tr>
<td>Kibana</td>
<td>Kibana access</td>
</tr>
<tr>
<td></td>
<td>Assign KIBANA_R role</td>
</tr>
<tr>
<td>MyPlug</td>
<td>MyPlug services access</td>
</tr>
</tbody>
</table>

**Children keys**

+ Add
2.6.6.3. Regenerate an API key

You can regenerate a key from the key edit screen by clicking the "Regenerate" button. A new key value will then be generated and the old value will be obsolete. If you regenerate a key (for example, if you lose it), you must update the systems that use it (MQTT devices, applications).

2.6.6.4. Disable / enable an API key

You may need to temporarily disable an API key. This is possible from the key edit screen by clicking on the "Disable" button. API access becomes impossible for any system using this key. Conversely, a key can be instantly reactivated by clicking on the "Enable" button present in the key details screen, or in the main list by clicking on the "Enable" icon.

2.6.6.5. Delete an API key

To delete one or more API keys, select them in the main list and click on the "Delete" icon.

The deletion is irreversible, and access to any system using the key(s) is impossible.

NB: Deleting a parent key will automatically delete all its children keys.

2.6.6.6. Put an API key in debug mode

Switching an API key to debug mode allows MQTT connection / disconnection information to be displayed in the activity log (see "Activity Log" section). Debug mode is disabled by default and can be enabled only for a limited time.

To put a key in debug mode, open the properties of the key by clicking on it, then click on the "Enable debug mode" button. A window will ask you to specify how long the debug mode will be active. At the end of this delay, the debug mode will be automatically disabled.
2.6.6.7. **Associate a certification authority**

You can authenticate your devices, or applications that are using an API key to connect to Live Objects using your own certificates. This is the principle of mutual authentication (SSL-two ways https://liveobjects.orange-business.com/doc/html/lo_manual_v2.html#go_further_and_use_2_way_authentication).

In addition to strengthening security, this functionality provides more flexibility in the management of MQTT device fleets, since it makes it possible to build device firmware with a single API key. Authentication is done on the basis of the device certificate.

To do this, you must first import into Live Objects the certificates of the certification authorities used to sign the client certificates of the applications or device that you want to authenticate. To do this, go to the "Configuration> API keys" menu. In the "CA certificate" tab click on the "import a CA certificate" button.
Then you can associate this certification authority with an API key. See the paragraph "Create an API key".

NB: If the certificate of your certification authority is deleted from Live Objects, the associated API keys will become inoperative. You will have to deactivate the "strong authentication" option. See the paragraph "Create an API key".

2.6.7. Manage firmwares

Firmware update requires that the device is connected with MQTT and implement the Live Objects "device management" logic (see the developer manual https://liveobjects.orange-business.com/doc/html/lo_manual_v2.html#MQTT_DEV_RSC)

To be able to update the firmware of your devices, you must first have created the corresponding firmware(s) in Live Objects. This consists of two steps:

1. Creation of the firmware with the same ID as that communicated by the device
2. Create the different versions attached to this firmware and import into Live Objects the corresponding binaries.

Step 1
To create a new firmware, go to « Firmwares » in the menu « Configuration » and click on the button “Create firmware”
Enter **the same firmware id as the one reported by your device**, and select the “http-updater” connector. Click the “Create a new firmware” button to complete step 1.

**Step 2**

In the main list, click on « Add version». 
Indicate the version's id in coherence with the one communicated by your device, then import the binary file corresponding to the firmware.

NB: You have the possibility to manage the compatibility between versions of a same firmware. To do this, enter the old versions of the firmware compatible with this new version in the "Compatible version" field. If you do not specify any version, this means that the new version can be installed on any previous version.

The available versions of a firmware are listed when clicking on the icon corresponding to the line of the firmware.
You can now program a firmware update. To do this, go to the “Devices” menu. (See section "Update the firmware of an MQTT device".

2.7. Alarms

In this menu you can set alarms and view them. The types of alarms currently available are:

- "Silent machine": This type of alarm allows detecting devices that no longer emits data. The device is then said "silent", which may be a sign of a malfunction (for example a problem related to the device itself, or local connectivity). The definition of "silent device", especially the period of inactivity, is done via "silent machine" rules.

In next versions of Live Objects, new types of alarms currently only available via API will be managed through the portal:

- Event Processing
- State processing

2.7.1. Display alarms

Go to “Alarms” menu, and select the type of alarm you want to display.

![Image of the Live Objects screen showing Silent machine alarms]

You can filter the list on the origin of the alarm (name of the rule or devices)

2.7.2. Set a silent machine rule

To create, modify or delete a silent machine rule, click on the sub menu “silent machine rules” of the “Alarms” menu.

![Image of the Live Objects screen showing Silent machine rules]

From this screen you can view your rules and their main characteristics. You can also suspend a rule which will change its status to “disabled”, reactivate it or permanently delete it.
Creating a rule is done in three steps:

First select the source to monitor

This can be a group of devices, or a specific device, or all the devices of your fleet. Once the source is defined, give it a name and specify the delay of inactivity that will trigger the alarm. In the example below, if a device included in the source selected in the previous step no longer sends a message for more than 10 hours, an alarm will be generated. The minimum idle time is 10 minutes.

Finally you can associate an action with an alarm. Possible actions are:

- Notify the alarm by SMS or email
- Push the alarm by http to an external server
- Send the alarm in a fifo for consumption in MQTT by an external application
You can combine all or part of these means of notification in a same rule.
You can also choose to no notify, in this case the alarm remains visible in the "Alarms" menu of the Live Objects portal.

The configuration of the different notifications (sms, email, push http and fifo) is the same as for the data routing (see description of the screens in the paragraph "Routing data")

Moreover you can decide to be notified again periodically if the device stays silent. In the examples above, you had a first alarm after 10 hours of inactivity, then you are notified again every hour if the device is still silent.

NB: The "silence" delay is taken into account only from the date of creation of the rule. Example: I create at time T a rule to report silent devices for more than 10 days. I will have the first alarms at the moment T + 10 days, even if at the time of the creation of the alarm I already have devices that are silent for more than 10 days.
3. Resources

A set of resources is available from your Live Objects portal: https://liveobjects.orange-business.com/#/faq

You will find:

- Practical answers to frequently asked questions (FAQ)
- Video tutorials giving you a quick start to the different features of Live Objects
- The complete development guide for Live Objects. This is the documentation for developers and integrators.
- A complete training kit for Live Objects
- The REST API documentation (swagger). This is an interactive documentation that allows you to manipulate the APIs with your Live Objects account
- Postman collections to facilitate use of APIs
- Code samples for devices (SDK) and for business applications (samples)
- Recommended tools for BI, analytics and dataviz that can be easily used with Live Objects
- An "About" section in which you can find the current version and its content (Release Note)

You can also join the Live Objects community on the developer forum https://stackoverflow.com/tags/orange-api