OneAPI Interface

Ref. 16051301 - Push SMS.

May 13, 2016 v1.1

Index

1 Introduction

This document explains the LabsMobile OneAPI Interface. It is designed for technicians and clients that want to integrate an application or software with LabsMobile messaging platform. The purpose of integration is to provide access to the SMS messaging services of LabsMobile.

This document contains a detailed explanation of the integration process. If you have any questions or need examples of code, please contact your usual LabsMobile agent or contact us at:

support@labsmobile.com

www.labsmobile.com

1.1 Changelog

v1.1 - 13/05/2016
- First version of the manual

2 OneAPI definition

2.1 OneAPI RESTFull Short Messaging Interface

The OneAPI Interface was created by the GSM Association and attempts to standardize communications between operators, integrators, etc. In this case we treat the case of the OneAPI Restfull that describe SMS services.

The OneAPI Restfull is a definition of methods to send and receive SMS messages, get the result of sent messages and receive/subscribe to change notifications of states of these messages.

2.2 Required information

For the integration with the OneAPI Interface the following information provided by LabsMobile is essential:

- Username and password (shown on the registration e-mail)
- Specific URL: *https://api.labsmobile.com/oneapi/2/smsmessaging/v1/*

Optionally you can communicate to LabsMobile the following parameters:

- IP Address where the messages will be sent. For security purposes the messaging platform will only admit messages from this/ these IP/s. This functionality is optional, this option is not activated by default and messages from any IP address will be accepted.
- Sender by default (default TPOA, by default is LABSMOBILE unless otherwise indicated).
- Messages daily limit, by default up to 50,000 sms/per day.
- Messages limit by batch, by default up to 10,000 sms/sending.

IMPORTANT: All these parameters will be assigned with the values by default to all accounts.

2.3 Tecnology and format

Communication with OneAPI is done with HTTP/POST calls with variables encoded in JSON format. Each function or method has a different URL and specific variables.

2.4 Authentication

The authentication method used is specified by the standard IETF RFC 2717. The platform verifies all calls to the API-Interface SMS HTTP before processing the sending. Also the HTTP call must be done from a valid IP (optional) and meet all account limitations and filters.

If the user name or password is not correct, the platform will respond with an HTTP 401 Unauthorized standard code.

If the call or HTTP request is from an IP not set as a valid source, the platform will respond with an HTTP 403 Forbidden standard code.

Example in PHP:

```
$ch = curl_init($url);
curl_setopt($ch, CURLOPT_HTTPAUTH, CURLAUTH_BASIC);
curl_setopt($ch, CURLOPT_USERPWD, $username.':'.$password);
curl_setopt($ch, CURLOPT_POST, true);
curl_setopt($ch, CURLOPT_POSTFIELDS, 'XmlData='.$sms);
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
curl_setopt($ch, CURLOPT_HEADER, true);
curl_setopt($ch, CURLOPT_TIMEOUT, 15);
$result = curl_exec($ch);
```

3 SMS push sending (MT)

Sending SMS messages is done in the following URL:

```
       https://api.labsmobile.com/oneapi/2/smsmessaging/v1/outbound/
tel:<sender_tel>/requests
```

Where *<sender_tel>* is the sender phone number. All variables have to be included in JSON format:

```
{"outboundSMSMessageRequest": {
       "address": {"tel:+31620445893”},
       "senderAddress": "tel:31620",
       "outboundSMSTextMessage": {
              "message": "Hello World",
       },
       "clientCorrelator" : "A23456654321",
       "receiptRequest": {
              "notifyURL": "http://application.example.com/notifications",
       },
       "senderName" : "SENDER”,
}}
```

Here is a list and description of the variables:

Name	Description
address	Required. st of phone numbers of the recipients. The format is always like: *tel.+<country_code><phone>*
senderAddress	Required. Sender of the message in numeric format.
message	Required. Message text. You can exceed 160 characters and then be interpreted as a SMSLong. You can send only characters supported by the GSM alphabet.
clientCorrelator	Optional. Sending identifier.
notifyURL	Optional. Link where confirmations and status changes of the sent messages will be received.
senderName	Optional. Sender of the message in alphanumeric format for those countries and operators that allow this feature.

3.1 Response messages and errors

Any method call to send SMS messages to the OneAPI will return a result in JSON format. This result will be returned in the body or stdout of the HTTP call.

The result of a call without any error is:

```
HTTP/1.1 200 Created
Content-Type: application/json
```

```
{"outboundSMSMessageRequest": {
       "address": {“tel:+31620445893”},
       "senderAddress": “tel:31620",
       "outboundSMSTextMessage": {
              "message": “Hello World",
       },
       "clientCorrelator" : “A23456654321",
       "receiptRequest": {
              "notifyURL": “http://application.example.com/notifications",
       },
       "senderName" : “SENDER”,
       "resourceURL" : “https://api.labsmobile.com/oneapi/2/smsmessaging/v1/
outbound/tel:<sender_tel>/requests/<subid>“,

}}
```

Where *resourceURL* is the link where there is a service to consult the status of the messages of this sending.

If the HTTP call generates an error, the response would be a HTTP code 400 (Bad Request) with the following format JSON:

```
HTTP/1.1 400 Bad Request
Content-Type: application/json

{"requestError": {
       "serviceException": {
              "messageId" : “SVC0001",
              "text" : “Service error occurred”,
       }
}}
```

Here is the list of possible errors:

Código	Descripción
SVC0001	Service error occurred
SVC0002	Invalid input value: message
SVC0004	No valid address(es)
SVC0005	Duplicate correlator

4 Receiving status changes

A client of the *OneAPI Interface* can ask to receive delivery confirmations on your system asynchronously. With the variable *NotifyURL* from the JSON format. Then all the changes of the messages will be notified to this link: status changes, delivery confirmations and errors.

IMPORTANT: some operators may not have available routes or any level of delivery confirmation.

The call to the provided url is a HTTP/POST call with JSON format variables in the body. The format would be like:

```
{"deliveryInfo": {
       "address" : "tel:+31620445893",
       "deliveryStatus" : "DeliveredToTerminal",
}}
```

The possible status are as follows:

Code
DeliveredToTerminal
DeliveryImpossible
DeliveryUncertain
MessageWaiting
DeliveredToNetwork