VIBER, SMS, LANDING PAGES & ONLINE BOOKING

OMNICHANNEL MESSAGING

Platform for Business

OCTAPUSH
OMNI MESSAGING PLATFORM

OMNI API DOCUMENTATION V1.7

yuboto

OMNI API Documentation

Contents

1. Getting Started with OMNI API

OMNI API services are a robust set of endpoints providing all the functionality necessary to effectively engage with your audience via SMS and Viber. This functionality is also available to end-users through the Octapush platform.

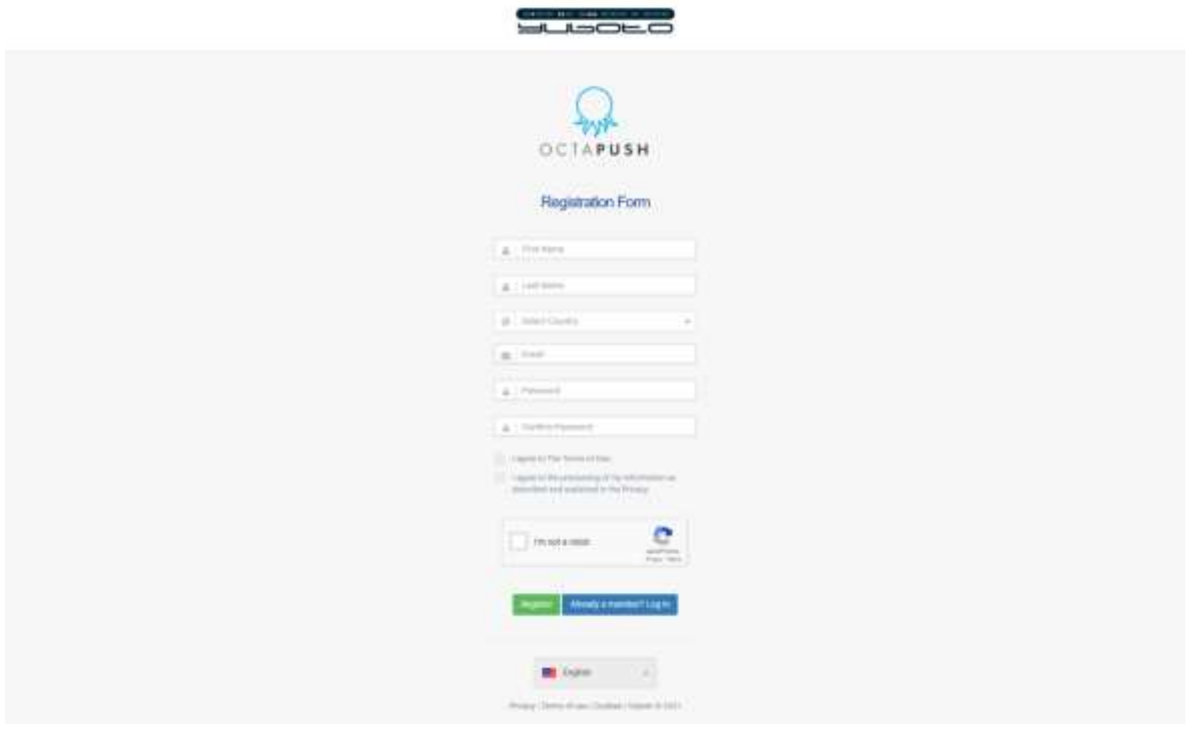
https://octapush.yuboto.com

OMNI API supports both HTTP POST and HTTP GET requests for all its endpoints. It can be incorporated into any existing infrastructure with the use of HTTP requests.

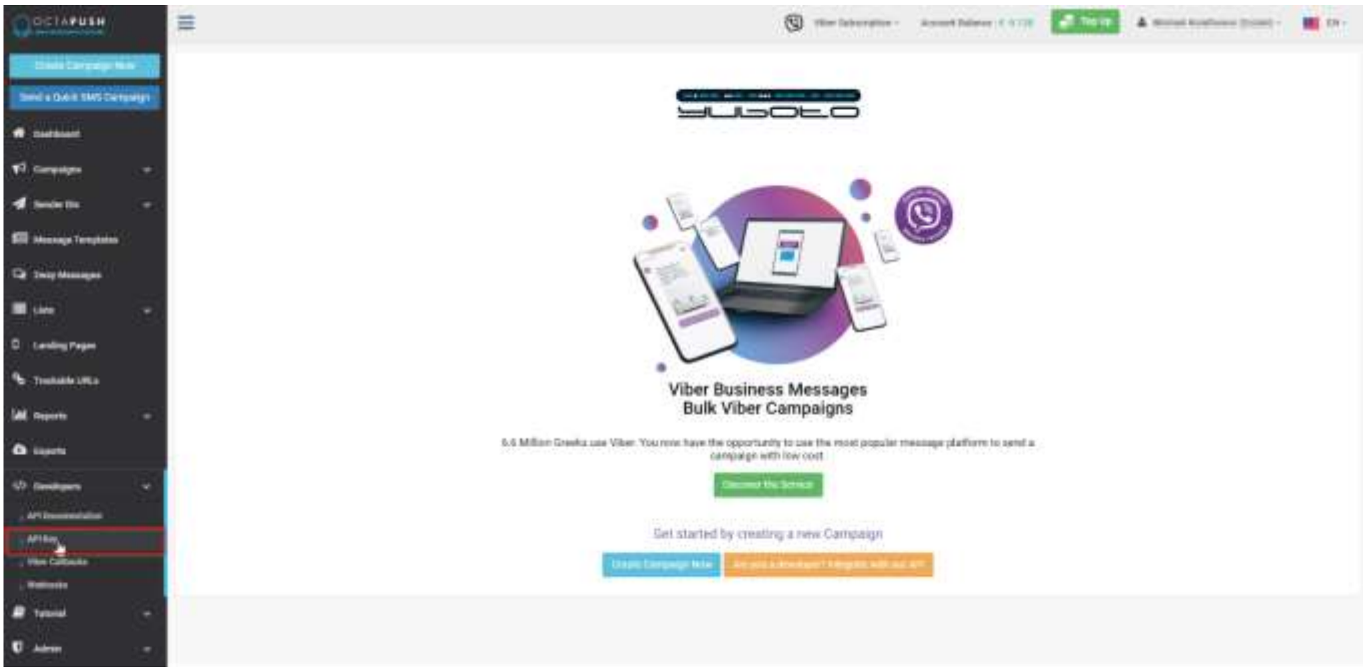
1.1. Activating the Services

The service usage requires an activated account that can be created at the Octapush platform.

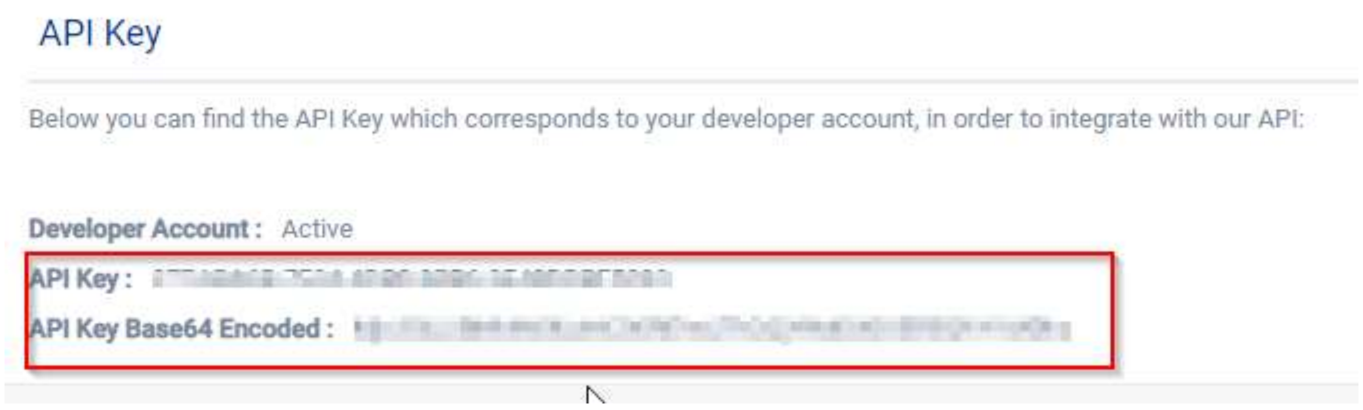
Follow the step required to register at https://octapush.yuboto.com/en-US/Register.



Once registered to the platform, log in to your new account at https://octapush.yuboto.com. Navigate to your account name on the platform screen, click on it and select "Developers" in the menu that appears or from the right menu click Developers and select API Key menu item.



Then copy the OMNI API Key found in the "API Key" section. This key will be needed for all transactions made to the endpoints. The key can also be provided by our support team at support@yuboto.com.



API Key

Below you can find the API Key which corresponds to your developer account, in order to integrate with our API:

Developer Account : Active

API Key : [redacted]

API Key Base64 Encoded : [redacted]

You will also need to top up your account balance or purchase the required credits to start sending messages.
There is an initial 0.10 euro free of charge for the initial tests. Beyond that point usage of the services requires enough funds to be available for each transaction.

The API key and the balance are required to send messages. All other services provided like Cost, subscription, etc. can be accessed with the use of the API Key even without any balance available.

It is critical to keep your API Key safe to prevent any unauthorized access. Once you obtain your API Key, you will have to use it in every API call.

If you need information about message charges or more information about Viber messages, please contact us at sales@yuboto.com

If you need technical information, please contact us at support@yuboto.com

You can also call us at +30 211 11 44 111 working days from 9.00 am to 6.00 pm.

The use of Yuboto platforms is subject to the terms of use and privacy statement you may find at https://octapush.yuboto.com.

1.2. OMNI API process flow

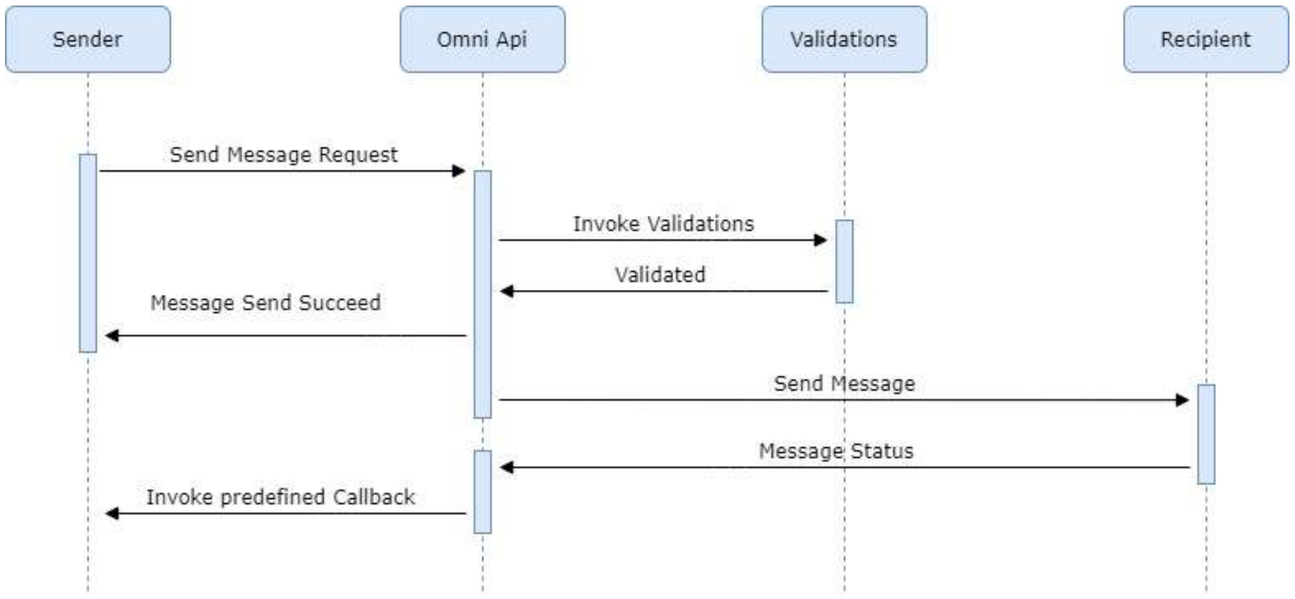
1.2.1. Generic process flow

When a message sent is requested to the API is a series of actions take place. Those actions are affected by the message channel (SMS, Viber), the message status (delivered, failed, etc.), and the sender presets such as Callback URL.

Once OMNI API receives the request it invokes a series of validations, such as whether there are invalid phone numbers, available account balances for the delivery, etc. The valid request is then sent to the recipient. The final status of the message will be pushed to a defined Callback URL if set in the message settings. The delivery reports are also available on the Octapush platform.

The message Callback (paragraph 1.4) responses carry a status (paragraph 2.2), and a set of information that is provided to the Callback URL.

Additionally, a callback can be invoked once the message is seen, clicked, etc.



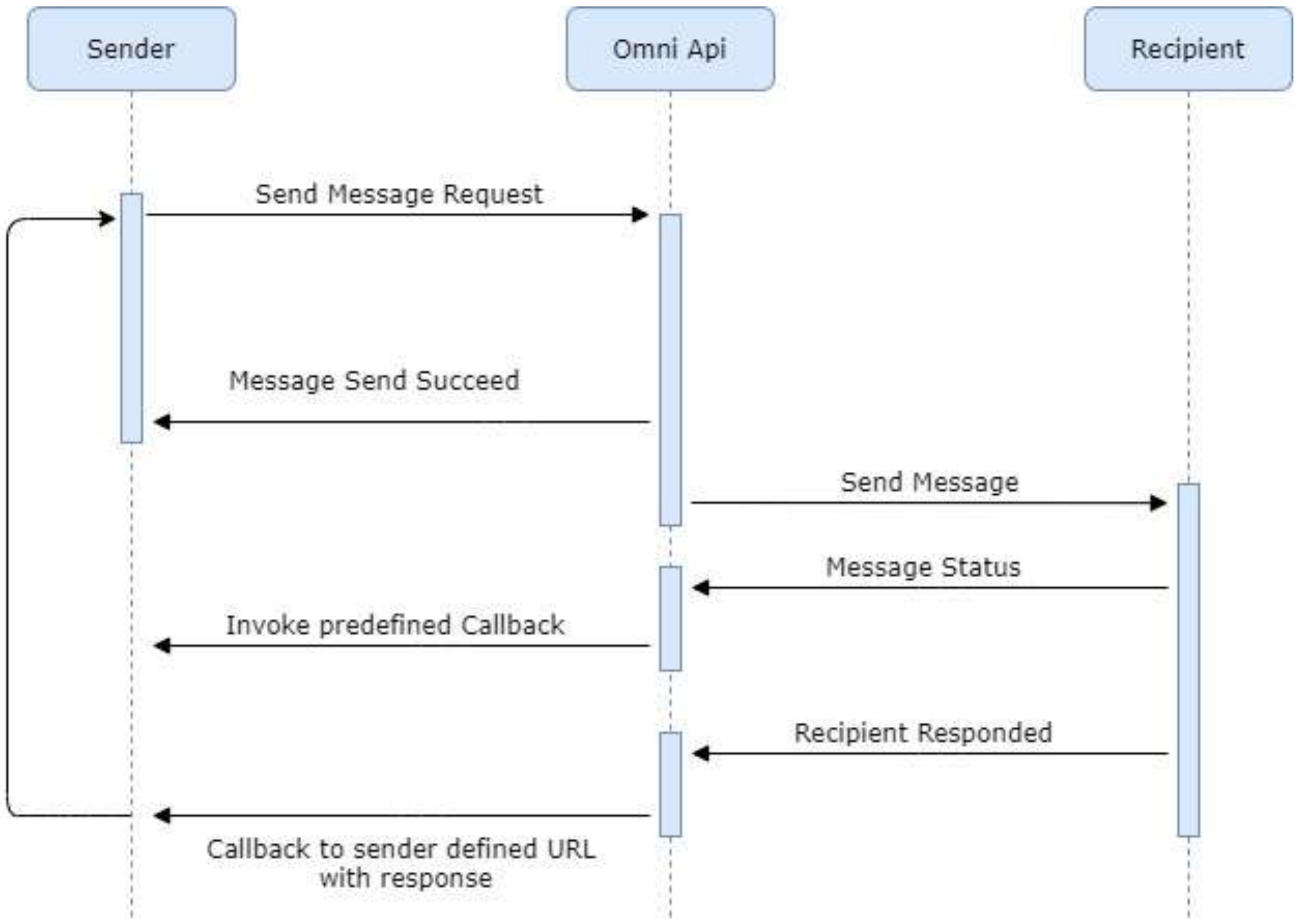
1.2.2. 2Way & Session messaging

2Way and Session messaging allow for two-way communication between the sender and the recipient of the message. These features can be activated in the Octapush platform.

Contact support@yuboto.com in order to set your persistent callback URL under your account for the recipient responses. Once a message is sent as a 2way or Session any reply will be also pushed to the specific URL. Otherwise, responses can also be viewed via the Octapush platform.

Then sender can then create a new send message request to reply, and the flow is again repeated.

Each communication sequence carries a distinct id "smsid". This id is produced every time a message is created by the sender and is included in all recipient direct responses to this message. Once the sender replies with a new message, another id is produced and again all replies based on this will carry this new id.



An example, for better understanding, could be a conversation between the company and its customer:

- Company's 1st Message:
(smsid: 06B6E2B5-E559-4936-AE69-C540F560F07E)

Welcome to Company's Viber Channel! Here you will find all our new offers and news.

- 1st Customer's Reply:
(smsid: 06B6E2B5-E559-4936-AE69-C540F560F07E)

Good evening, is it possible to buy with bank deposit as a payment method?

- Company's Response to "1st Customer's Reply":
(smsid: 0EA8F98A-F77D-404A-B1B6-4D180E1C25A8)

Good evening, Of course! Which bank you will prefer?

- 2nd Company's Customer Reply:
(smsid: 0EA8F98A-F77D-404A-B1B6-4D180E1C25A8)

Is there a possibility for Alpha Bank?

- Additional message as 2nd Company's Customer Reply:
(smsid: 0EA8F98A-F77D-404A-B1B6-4D180E1C25A8)

Or Piraeus?

- Company's Response to "2nd Company's Customer Reply":
(smsid: C4AA1C08-B639-428B-901D-D2CB7FE6E337)

Yes. The Alpha Bank IBAN is GR33333441124545.

- 3rd Company's Customer Reply:
(smsid: C4AA1C08-B639-428B-901D-D2CB7FE6E337)

Thank you!

It is important to notice on the example above how the "smsid" is used to correlate between messages sent and their respective replies. Each message send to a recipient will generate the "smsid". When the recipient replies the system forwards the reply along with the "smsid" that the reply is based upon.

The same path is followed for the cases that the response includes a file. The recipient response is forwarded to the predefined callback URL along with a link to the file (paragraph 1.4.3). The file resides on the server for 7 days since the recipient's response.

1.2.3. Session messaging

A session message is like a 2way message but with different charging scheme. Once a message is sent as a Session message the initial charge is as a 2way. If the recipient responds to the initial message, then automatically charging changes to a Session charge. For as long as Session messaging is active no additional charges occur. The limitations in such a case are as follow:

A session may last up to 12 hours.

During a session, a sender can send up to 60 messages.

A sender can send up to 10 consecutive messages to a recipient without the recipients reply.

In any other case a new session will begin.

2. Base URL

All the requests are submitted through **HTTP POST or HTTP GET** methods.

Base URL: https://services.yuboto.com/

Service Endpoints at: /omni/v1

Using the OMNI service, you can perform the following actions:

- **Send:** Send SMS or Viber messages (Paragraph 1)
- **Dlr:** Retrieve the status of previously sent messages (Paragraph 2)
- **Cost:** Request the cost of SMS and Viber messages (Paragraph 3 and Paragraph 4)
- **Balance:** Request your account’s balance (Paragraph 5)
- **Subscription:** Monitor your subscription information (Paragraph 6)
- **Cancel:** Cancel scheduled messages (Paragraph 7)
- **Create Key:** Create an API Key for your subaccounts (Paragraph 8)
- **Two Way Authentication Validation**: Validate the pin for a specific smsid for two-factor authentication messages (Paragraph 9).
- **Phone Number Reporting**: Retrieve all existing send/delivery reports for previous campaigns based on the recipient’s phone number (Paragraph 10)
- **Subscriber Lists**: Create and modify contact lists (Paragraph 11).
- **Contacts**: Create and modify contacts within the subscriber lists (Paragraph 12).
- **Blacklists**: Manage the list of suspended from usage phone numbers (Paragraph 13).

3. HTTP POST Request

3.1. Content-Type

The requests must be in JSON format and require an HTTP header “Content-Type” with a value “application/json; charset=utf-8”.

3.2. Authentication

All POST API calls require authentication. This is essential for the API to identify which user is making the call so that appropriate results will be returned, as well as for security reasons.

For this purpose, API uses basic authentication. Authentication data are sent via the HTTP header “Authorization”.

Steps to construct authorization header:

1. Base64 encode the API Key (or use the encoded equivalent provided in the Octapush platform*).
2. Supply an "Authorization" header with content "Basic" followed by the encoded API Key. For example, the Authorization header will be:

```
Authorization: Basic apiKey
```

*Find your OMNI API Key in the "My Account" screen at the "API Integration" section which is located on your Octapush account or request it from our support team at support@yuboto.com. It is critical to keep your API Key safe to prevent any unauthorized access. Once you obtain your API Key, you will have to use it in every API call you make.

4. HTTP GET Request

4.1. Parameters

In order to populate the Data Objects when using GET requests, the use of GET parameters is required. Since the data objects contain **Complex Types**, the use of the '.' characters are used to set a property of an Object. For example, for the SMS's text to be set, the following expression should be used, sms.text={{Text}}.

An example request is the following:

```
https://services.yuboto.com/omni/v1/Send?phonenumbers=3069XXXXXXXX&sms.sender
=TestSender&sms.text=test%20message
```

4.2. Authentication

All GET API calls require authentication. This is essential for the API to identify which user is making the call so that appropriate results will be returned, as well as for security reasons.

For this purpose, API uses 2 different authentication methods.

4.2.1. Header Authentication

Authentication data are sent via the HTTP header "Authorization".

Steps to construct authorization header:

1. Base64 encode the API Key (or use the encoded equivalent provided in the Octapush platform*).

2. Supply an "Authorization" header with content "Basic" followed by the encoded API Key. For example, the Authorization header will be:

```
Authorization: Basic apiKey
```

Example:

```
https://services.yuboto.com/omni/v1/Send?phonenumbers=3069XXXXXXXX&sms.sender
=TestSender&sms.text=test%20message
```

*Find your OMNI API Key in the "My Account" screen at the "API Integration" section which is located on your Octapush account or request it from our support team at support@yuboto.com. It is critical to keep your API Key safe to prevent any unauthorized access. Once you obtain your API Key, you will have to use it in every API call you make.

4.2.2. URL Authentication

Authentication data are sent via GET Parameter "apiKey".

1. Base64 encode the API Key (or use the encoded equivalent provided in the Octapush platform*).
2. Add the API Key with as a parameter "apiKey" to each query:

apiKey Parameter Authentication Example:

```
https://services.yuboto.com/omni/v1/Send?phonenumbers=3069XXXXXXXX&sms.sender
=TestSender&sms.text=test%20message&apiKey=XFv4RERCQTktNjBEQi00ODAyLUE3NTEtMT
UyMDkzMsU4QkdFL
```

*Find your OMNI API Key in the "My Account" screen at the "API Integration" section which is located on your Octapush account or request it from our support team at support@yuboto.com. It is critical to keep your API Key safe to prevent any unauthorized access. Once you obtain your API Key, you will have to use it in every API call you make.

REFERENCE

1. Send Method

Description

This method allows you to send text messages to one or multiple recipients simultaneously. The maximum number of recipients you can send at one time is 1000.

When the callbackUrl property is defined the system will call the URL for each message status.

URL to web service operation

```
https://services.yuboto.com/omni/v1/Send
```

1.1. Method Parameters, Request, and Response

Request Parameters

The variables used to send text messages (SMS and/or Viber) are:

Variables	Description	Permitted Values	Required
phonenumbers	Refers to the phone number of the recipient or recipients of the text message (use array for multiple recipients).	String/Array. Use country code without + or 00.	Yes
dateinToSend	Indicates the date you wish to send the message. If this is omitted, the message is sent instantly.	Integer. YYYYMMDD YYYY refers to the year MM refers to the month DD refers to the day	No
timeinToSend	Indicates the time you wish to send your message. If this is omitted, the message is sent instantly.	Integer. HHMM HH refers to the hour MM refers to minutes	No
dlr	The flag indicates if delivery receipt request must be sent to customer's application. (Default: false)	Bool	No
callbackUrl	When the message reaches its final state, a call to this url will be performed by Yuboto's system with the message's delivery info. See paragraph 1.4.	String	No***
option1	User defined value that will be included in the call to the provided callback_url.	String	No**
option2	User defined value that will be included in the call to the provided callback_url.	String	No**
sms	This object is required if the list of channels contains SMS channel.	SmsObj Object	No*
viber	This object is required if the list of channels contains VIBER channel. Parameters text, buttonCaption + buttonAction and image make Viber Service Message content. There are 5 possible combinations of Viber Service Message content: • text only, • image only, • text + button, • text + button + image • file • Video • Video + text • Video + text + button	ViberObj Object	No*

** One of two these parameters must always exist.*
***Option1 & Option2 Parameters will be available for retrieve only if you pass dlr:true and a callbackUrl parameter*
****You can add a persistent callback URL in your account without sending a callbackUrl parameter each time you call the API. Contact support@yuboto.com in order to set your persistent callback URL under your account.*

SmsObj parameters

Variables	Description	Permitted Values	Required
sender	SMS originator ("sender") that will be displayed on the mobile device's screen. • Alphanumeric origin, max. 11 characters • Numeric origin, max. 20 characters	String	Yes
Text*	The text of the message. If two-factor authentication is activated TwoFa, is mandatory that '**{pin_code}**' is included in this string. This placeholder will then be replaced with the generated pin.	String	Yes
validity	If the SMS is not delivered directly, this variable indicates the number of minutes for which the message will remain active, before being rejected by the SMSC.	Integer. Min Value: 30 Max Value: 4320 (3 Days) default: 1440 (1 Day)	No
typesms	Indicates the type of message you wish to send.	String. 1. sms (default) 2. Flash 3. unicode	No
longsms	Indicates if the message can be over 160 characters. It applies only to standard type SMS.	Bool. 1. false (default) 2. true	No
priority	Indicates which channel has priority when it comes to OMNI messaging (default value is: 0).	Integer	No
TwoFa	If Two-Factor Authentication is needed, then provide this object along with other values.	Object	No
FallbackOnFailed	Options for fallback activation when fallback action is enabled.	Object	No

ViberObj parameters

Variables	Description	Permitted Values	Required
sender	Viber message originator ("sender") that will be displayed on the mobile device's screen. • Alphanumeric origin, max. 28 characters	String	Yes

Text*	The Viber Service Message text. Length can be up to 1000 characters. VIBER text can be sent alone, without a button or image. When factor authentication is activated TwoFa, is mandatory and '{pin_code}' must be included in the string. This placeholder will be replaced with the generated pin.	String	No
validity	If the Viber message is not delivered directly, this variable indicates the number of seconds for which the message will remain active, before being rejected.	Integer. Min Value: 30*** Max Value: 1.209.600 (14 days) Default 86400 (1 day)	No
expiryText	Relevant for the iOS version of Viber application (iPhone users only). This is the text that will be displayed if Viber Service Message expires.	String	No
buttonCaption*	A textual writing on the button. The maximum length is 30 characters. The VIBER button can be sent only if Viber Service Message contains text.	String	No
buttonAction*	The link of button action. The maximum length is 255 characters. (Set as the URL of the file when sending a file message or Set as the URL of the video when sending Video, Video & Text or Video, Text & Button).	String	No**
Image*	The URL address of image sent to end-user. The VIBER image can be sent only alone or together with text and button.	String	No
filetype*	The type of file to be sent. Allowed types: Documents: doc, docx, rtf, dot, dotx, odt , odf, fodt, txt, info PDF: pdf, xps, pdax, eps Spreadsheet: xls, xlsx, ods, fods, csv, xlsm, xltx	String	No**
filename*	The name of the file to be downloaded. Maximum length 25 characters.	String	No**
duration	The duration of the video in seconds. Max value is 600 seconds	Integer	No****
fileSize	The size of the video in MB. Max value is 200MB	Integer	No****
thumbnail	The URL address of image sent as thumbnail of the Video. Max length of the URL is 1000 characters	String	No****
priority	Indicates which channel has priority when it comes to OMNI messaging (default value is: 0).	Integer	No
TwoFa	If Two-Factor Authentication is needed then provide this object along with other values.	Object	No
serviceType	Indicated the service type of the request.	Integer 1: OneWay,	No

		2: TwoWay, 3: Session	
FallbackOnFailed	Options for fallback activation when fallback action is enabled.	Object	No

* Parameters text, buttonCaption + buttonAction, and image make Viber Service Message content. There are 8 possible combinations of Viber Service Message content:

- Text Only
- Image Only
- Text & Button
- Text, Button & Image
- File
- Video
- Video & Text
- Video, Text & Button

** Required when sending file message.

*** Recommended minimum for TTL from Viber is 60 seconds.

**** Required when sending Video message

TwoFaObj parameters

Variables	Description	Type	Required
pinLength	The length of the pin to be generated. Min:4 Max: 32	int	Yes
pinType	Accepted values: • ALPHA (PQRST) • ALPHA_ALPHA_LOWER_NUMERIC (Pg3Gh) • ALPHA_NUMERIC (hEQsa) • NUMERICWITHOUTZERO (5443) • NUMERIC (54034)	String	Yes
isCaseSensitive	Whether the pin should be case sensitive (alpha, alphanumeric). If false, the case sensitivity would not be checked when validating, if true, the code for validation needs to be entered exactly as provided.	bool	Yes

expiration	The time the pin will be active. Accepted values between 60-600 (in seconds).	int	Yes

FallbackOnFailed provides a set of options that define the conditions when a fallback to a different channel occurs. When a condition is set to true is gets activated and the systems checks if this condition is met a fallback action will be invoked to the next defined channel. When a condition is set a false its gets inactive, and the system does not check whether this condition is met.

FallbackOnFailed parameters

Variables	Description	Type	Required
notDelivered	If message not delivered.	Boolean Default: true	Yes
userBlocked	If user blocked.	Boolean Default: true	Yes
expired	If message is expired.	Boolean Default: true	Yes
error	On Error.	Boolean Default: true	Yes
rejected	On message rejected.	Boolean Default: false	Yes

notDelivered: Message is not delivered to the recipient.

userBlocked: The user blocks receiving messages from the sender.

expired: The message has not yet been delivered within the predefined period.

error: A send error occurs.

Rejected: The message is rejected, as an example erroneous recipient.

Request Example (POST)

```
{
      "dlr": "false",
      "callbackUrl": null,
      "option1": null,
      "option2": null,
      "phonenumbers":"3069XXXXXXXX",
      "dateinToSend": null,
      "timeinToSend": null,
      "viber":{
            "sender": "Test 2way",
            "text": "This is a test viber message",
            "image": null,
            "buttonAction":null,
            "buttonCaption":null,
            "fileName":null,
            "fileType":null,
            "validity":180,
            "expiryText":"This viber message expired",
            "priority":0,
            "twofa":{
                  "pinLength":5,
                  "pinType":"NUMERIC",
                  "isCaseSensitive":"false",
                  "expiration":180
            },
            "fallbackOnFailed":{
                  "notDelivered":"true",
                  "userBlocked":"true",
                  "expired":"true",
                  "error":"true",
                  "rejected":"true"
            }
      },
      "sms":{
            "sender":"TestSender",
            "text":" This is a test sms fallback",
            "validity":180,
            "typesms":"sms",
            "longsms":"false",
            "priority":1,
            "twofa":{
                  "pinLength":5,
                  "pinType":"NUMERIC",
                  "isCaseSensitive":"false",
                  "expiration":180
            }
      }
}
```

Request Example Multiple Recipients (POST)

```
{
      "dlr": "false",
      "callbackUrl": null,
      "option1": null,
      "option2": null,
      "phonenumbers": ["3069XXXXXXXX", "3069XXXXXXXX"],
      "dateinToSend": null,
      "timeinToSend": null,
      "viber":{
            "sender": "Test 2way",
            "text": "This is a test viber message",
            "image": null,
            "buttonAction":null,
            "buttonCaption":null,
            "fileName":null,
            "fileType":null,
            "validity":180,
            "expiryText":"This viber message expired",
            "priority":0,
            "twofa":{
                  "pinLength":5,
                        "pinType":"NUMERIC",
                        "isCaseSensitive":"false",
                        "expiration":180
                },
                "fallbackOnFailed":{
                        "notDelivered":"true",
                        "userBlocked":"true",
                        "expired":"true",
                        "error":"true",
                        "rejected":"true"
                }
        },
        "sms":{
                "sender":"TestSender",
                "text":" This is a test sms fallback",
                "validity":180,
                "typesms":"sms",
                "longsms":"false",
                "priority":1,
                "twofa":{
                        "pinLength":5,
                        "pinType":"NUMERIC",
                        "isCaseSensitive":"false",
                        "expiration":180
                }
        }
}
```

Request Example (GET)

1- Header Authentication:

```
https://services.yuboto.com/omni/v1/Send?dlr=false&phonenumbers=3069XXXXXXXX&
viber.sender=Test%202way&viber.text=This%20viber%20message%20expired&viber.va
lidity=180&viber.expiryText=This%20viber%20message%20expired&viber.priority=0
&viber.twofa.pinLength=5&viber.twofa.pinType=NUMERIC&viber.twofa.isCaseSensit
ive=false&viber.twofa.expiration=180&sms.sender=TestSender&sms.text=This%20is
%20a%20test%20sms%20fallback&sms.validity=180&sms.typesms=sms&sms.longsms=fal
se&sms.priority=1&sms.twofa.pinLength=5&sms.twofa.pinType=NUMERIC&sms.twofa.i
sCaseSensitive=false&sms.twofa.expiration=180
```

2- URL Authentication (apiKey Parameter Authentication Example):

```
https://services.yuboto.com/omni/v1/Send?dlr=false&phonenumbers=3069XXXXXXXX&
viber.sender=Test%202way&viber.text=This%20viber%20message%20expired&viber.va
lidity=180&viber.expiryText=This%20viber%20message%20expired&viber.priority=0
&viber.twofa.pinLength=5&viber.twofa.pinType=NUMERIC&viber.twofa.isCaseSensit
ive=false&viber.twofa.expiration=180&sms.sender=TestSender&sms.text=This%20is
%20a%20test%20sms%20fallback&sms.validity=180&sms.typesms=sms&sms.longsms=fal
se&sms.priority=1&sms.twofa.pinLength=5&sms.twofa.pinType=NUMERIC&sms.twofa.i
sCaseSensitive=false&sms.twofa.expiration=180&apiKey=XFv4RERCQTktNjBEQi00ODAy
LUE3NTEtMTUyMDkzMsU4QkdFL
```

Response Example

```
{
    "ErrorCode":0,
    "ErrorMessage":"",
    "Message":[
        {
            "id":"MessageID1",
            "channel":"sms",
            "phonenumber":"306936XXXXXX",
            "status":"Submitted"
        },
        {
            "id":"MessageID2",
            "channel":"sms",
            "phonenumber":"306936XXXXXXX",
            "status":"Submitted"
        }
    ]
}
```

ErrorCode: The response error code for this call. This will be 0 if successful.

ErrorMessage: The response error message. This will be null if successful.

Message: A list that contains the status of the messages.

- **id:** The id of message status.
- **channel:** The channel that the message will be sent (SMS or Viber).
- **phonenumber:** Refers to the phone number of the recipient of the text message.
- **status:** The status of the message.

If this method called successfully then the ErrorCode has the value 0 and the ErrorMessage contains a zero-length string. If an error occurred then, consult the error message that appears.

1.1.1 Request (POST) Examples

SMS Request Example (POST)

```
{
      "dlr": "false",
      "callbackUrl": null,
      "option1": null,
      "option2": null,
      "phonenumbers":"3069XXXXXXXX",
      "dateinToSend": null,
      "timeinToSend": null,
      "sms":{
            "sender":"TestSender",
            "text":" This is a test sms fallback",
            "validity":180,
            "typesms":"sms",
            "longsms":"false",
            "priority":1
      }
}
```

Viber Request Example (POST)

```
{
      "dlr": "false",
      "callbackUrl": null,
      "option1": null,
      "option2": null,
      "phonenumbers":"3069XXXXXXXX",
      "dateinToSend": null,
      "timeinToSend": null,
      "viber":{
            "sender": "Test 2way",
            "text": "This is a test viber message",
            "image": null,
            "buttonAction":null,
            "buttonCaption":null,
            "fileName":null,
            "fileType":null,
            "validity":180,
            "expiryText":"This viber message expired",
            "priority":0
      }
}
```

Viber with fallback to SMS

```
{
      "dlr": "false",
      "callbackUrl": "https://my.exampleCallbackUrl.com/webhook",
      "option1": null,
      "option2": null,
      "phonenumbers":"3069XXXXXXXX",
      "dateinToSend": null,
      "timeinToSend": null,
      "viber":{
            "sender": "Test 2way",
            "text": "This is a test viber message",
            "image": null,
            "buttonAction":null,
            "buttonCaption":null,
            "fileName":null,
            "fileType":null,
            "validity":180,
            "expiryText":"This viber message expired",
            "priority":0,
            "fallbackOnFailed":{
                  "notDelivered":"true",
                  "userBlocked":"true",
                  "expired":"true",
                  "error":"true",
                  "rejected":"true"
            }
```

```
        },
        "sms":{
                "sender":"TestSender",
                "text":" This is a test sms fallback",
                "validity":180,
                "typesms":"sms",
                "longsms":"false",
                "priority":1,
        }
}
```

1.1.2 Request (POST) Examples of SmsObj & ViberObj for each message type

SmsObj for Text Sms message (POST)

```
        "sms":{
                "sender":"TestSender",
                "text":" This is a test sms fallback",
                "validity":180,
                "typesms":"sms",
                "longsms":"false",
                "priority":1
        }
```

SmsObj for Two Factor (pin) authentication (POST)

```
        "sms":{
                "sender":"TestSender",
                "text":" This is a test sms with your {pin_code}",
                "validity":180,
                "typesms":"sms",
                "longsms":"false",
                "priority":1,
                "twofa":{
                        "pinLength":5,
                        "pinType":"NUMERIC",
                        "isCaseSensitive":"false",
                        "expiration":180
                }
        }
```

ViberObj for Text only message (POST)

```
        "viber":{
                "sender": "Test 2way",
                "text": "This is a test viber message",
                "image": null,
                "buttonAction":null,
                "buttonCaption":null,
                "fileName":null,
                "fileType":null,
                "validity":180,
                "expiryText":"This viber message expired",
                "priority":0,
        }
```

ViberObj with fallback options message (POST)

```
"viber":{
        "sender": "Test 2way",
        "text": "This is a test viber message",
        "image": null,
        "buttonAction":null,
        "buttonCaption":null,
        "fileName":null,
        "fileType":null,
        "validity":180,
        "expiryText":"This viber message expired",
        "priority":0,
        "fallbackOnFailed":{
                "notDelivered":"true",
                "userBlocked":"true",
                "expired":"true",
                "error":"true",
                "rejected":"true"
        }
}
```

ViberObj for Image only message (POST)

```
"viber":{
        "sender": "Test 2way",
        "text": null,
        "image": "https://www.myimageurl.com/myimage.jpg”,
        "buttonAction": null
        "buttonCaption": null,
        "fileName": null,
        "fileType": null,
        "validity":180,
        "expiryText":"This viber message expired",
        "priority":0
}
```

ViberObj for Text & Action Button message (POST)

```
"viber":{
        "sender": "Test 2way",
        "text": "This is a test viber message with action button",
        "image": null,
        "buttonAction":"https://www.mybuttonaction.com”,
        "buttonCaption":"click me and go to my action”,
        "fileName":null,
        "fileType":null,
        "validity":180,
        "expiryText":"This viber message expired",
        "priority":0,
}
```

ViberObj for Text & Image & Action Button message (POST)

```
"viber":{
```

```
            "sender": "Test 2way",
            "text": "This is a viber message with text, image and action button",
            "image": "https://www.myimageurl.com/myimage.jpg",
            "buttonAction":"https://www.mybuttonaction.com",
            "buttonCaption":"click me and go to my action",
            "fileName":null,
            "fileType":null,
            "validity":180,
            "expiryText":"This viber message expired",
            "priority":0
      }
```

ViberObj for File message (POST)

```
      "viber":{
            "sender": "Test 2way",
            "text": null,
            "image": null,
            "buttonAction":"https://www.myfilepath.com/myfile.doc",
            "buttonCaption": null,
            "fileName":" myfile.doc",
            "fileType":"doc",
            "validity":180,
            "expiryText":"This viber message expired",
            "priority":0
      }
```

ViberObj for Video message (POST)

```
      "viber":{
            "sender": "Test 2way",
            "image": null,
            "buttonAction":"https://videourladdress.com/video.mp4",
            "buttonCaption":null,
            "duration":30,
            "fileSize":10,
            "thumbnail": "https://www.myimageurl.com/myimage.jpg",
            "validity":180,
            "expiryText":"This viber message expired",
            "priority":0
      }
```

ViberObj for Video & Text message (POST)

```
      "viber":{
            "sender": "Test 2way",
            "image": null,
            "text": "This is a viber message with video and text",
            "buttonAction":"https://videourladdress.com/video.mp4",
            "buttonCaption":null,
            "duration":30,
            "fileSize":10,
            "thumbnail": "https://www.myimageurl.com/myimage.jpg",
            "validity":180,
            "expiryText":"This viber message expired",
            "priority":0
      }
```

ViberObj for Video, Text & Button message (POST)

```
	"viber":{
		"sender": "Test 2way",
		"image": null,
		"text": "This is a viber message with video, text and button",
		"buttonAction":"https://videourladdress.com/video.mp4",
		"buttonCaption":"click me and go to video”,
		"duration":30,
		"fileSize":10,
		"thumbnail": "https://www.myimageurl.com/myimage.jpg”,
		"validity":180,
		"expiryText":"This viber message expired",
		"priority":0
	}
```

ViberObj for Two Factor (pin) authentication (POST)

```
	"viber":{
		"sender": "Test 2way",
		"text": "Your pin number for this viber message is {pin_code}",",
		"image": null,
		"buttonAction":null,
		"buttonCaption":null,
		"fileName":null,
		"fileType":null,
		"validity":180,
		"expiryText":"This viber message expired",
		"priority":0
	}
```

1.2. Type of SMS Messages

Through Yuboto API, you can send:

- Simple SMS (up to 160 characters)
- Flash SMS (up to 160 characters)
- Unicode SMS (up to 70 characters)
- Long SMS (more than 160 7bit characters or 70 Unicode characters)

A simple SMS includes all the 7bit alphabet characters as defined by GSM 03.38.

Some 8bit alphabet characters may also be included and sent as a simple SMS. These will count as 2 characters.

These characters are:

CIRCUMFLEX ACCENT	^
LEFT CURLY BRACKET	{

RIGHT CURLY BRACKET	}
REVERSE SOLIDUS (BACKSLASH)	\
LEFT SQUARE BRACKET	[
TILDE	~
RIGHT SQUARE BRACKET	]
VERTICAL BAR	|
EURO SIGN	€

All the other characters included in the 8bit alphabet can only be sent as Unicode characters (SMS 70 characters).

For more information about Unicode characters, you can visit http://www.unicode.org/charts/.

If you use small case Greek characters (8bit) in a non-Unicode format, then the system will automatically convert them into Capital Greek characters (7bit).

Long SMS is a text message longer than 160 characters or 70 characters for Unicode type.

Long SMS (more than 160 characters for 7bit messages and more than 70 characters for 8bit messages) will be concatenated, ensuring that a multi-part message is received as a single SMS message. This avoids the confusion of messages arriving in random order. This is accomplished through User Data Header (UDH) which is a binary structure at the start of a SMS. It specifies how the message should be formatted and processed and holds 7 characters of each message part.

If the end user's mobile phone not supporting long SMS, then the message will be divided into multiple messages of 153 characters for 7bit SMS or 63 characters for 8bit SMS each (Maximum number of characters 2000).

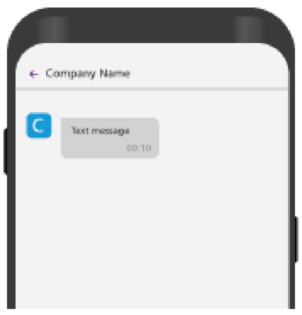
If you choose to send a long SMS without previously notifying the system, then the system will limit it to 160 characters (simple SMS).

1.3. Type of Viber Messages

Through Yuboto API, you can send:

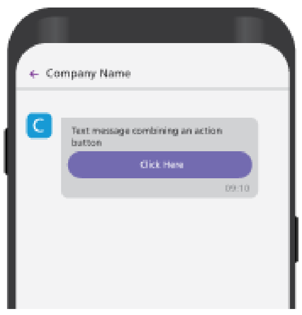
- Text Only
- Image Only
- Text & Action Button
- Text, Image & Action Button
- File
- Video
- Video & Text
- Video, Text & Button

Text Only



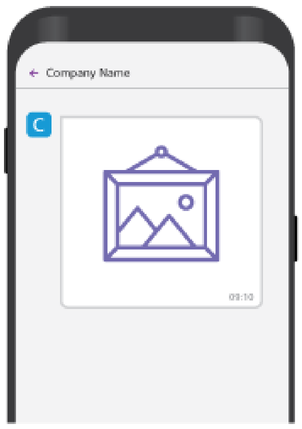
Text message

Text & Action Button

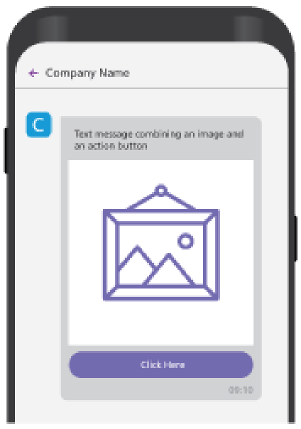


Text message combining an action button

Image Only

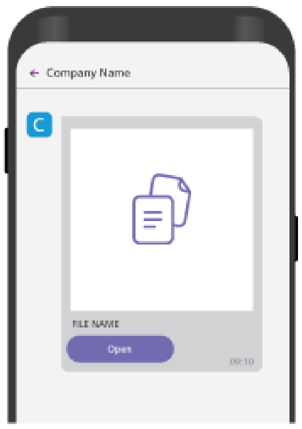


Text, Image & Action Button

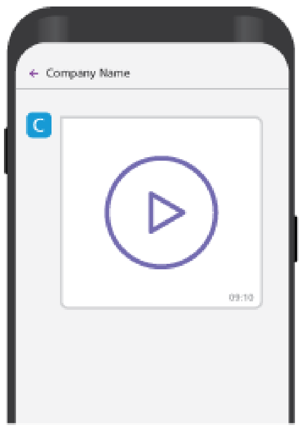


Text message combining an image and an action button

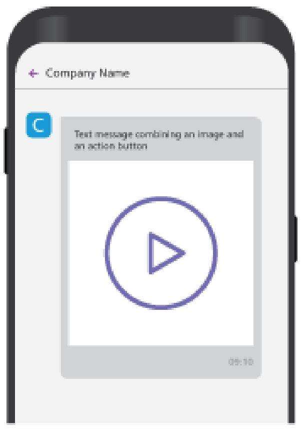
File



Video Only

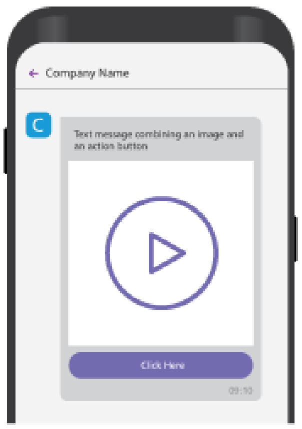


Video & Text



Text message combining an image and an action button

Video, Text & Action Button



Text message combining an image and an action button

1.4 System Forwarded Callbacks

When callback is true (defined in the Octapush platform), Yuboto's system will forward the final state info to the client. The information that the client will receive, it is possible for the user to pass it on dynamically to the system.

The message info will be forwarded through a get request to the callback URL. The following parameters will be included in the query string:

Parameter	Type	Description
sender	string	The message sender
receiver	string	The destination phone number
smsid	string	The unique id that the message has
status	int	The status code that the message has
statusDescription	string	See the provided table for a detailed description of status values
dlrDate	String (YYYY-MM-DD HH:MM:SS)	The date that the message delivered
channel	string	The channel with which the message was sent
option1	string	The user-defined value that was passed to method 'send'
option2	string	The user-defined value that was passed to method 'send'

Note: In case the callback is false, then Yuboto's system will not send to the client the final state info.

1.4.1 Status Expired Callback

An expired callback will be sent on iOS once the expiry text message is seen. On Android, the expired callback will be sent once the user is online again.

1.4.2 Message Response Callback

In the cases of 2way & Session messaging the end-user may respond. This response can be sent to a predefined callback URL. A response callback example:

```
https://www.example.com/example?receiver=3069XXXXXXXX&dlrDate=2021-01-8
10:13:21&sender=Test%202way&userMessage=Test%20response%20message&smsid=xxxxx
xxx
```

Parameter	Type	Description
sender	string	The message sender
receiver	string	The destination phone number
smsid	string	The unique id that the message has
dlrDate	String (YYYY-MM-DD HH:MM:SS)	The date that the message delivered
userMessage	string	The message of the end user's response

If additional callback data are predefined, then they will be appended to the query.

1.4.3 File Response Callback

In the cases of 2way & Session messaging the end-user may respond with a file. The response can be sent to a predefined callback URL. The file is kept on the server for 7 days. A response callback example:

```
https://www.example.com/example?receiver=3069XXXXXXXX&dlrDate=2021-01-8
10:13:21&sender=Test%202way&downloadUrl=http://example.com/DownloadUserFile/?
id=EAAAAO8LzuRF%2fEUV2UxZeH0BEK3FI3H1an3byKhHkrtuafKeuArzJRzmRrJMBPS0QGs86ANV
J2hXRP7P3d%2biELhpmUY%3d&smsid=E0CFD97F-01CB-4F82-8B8E-
E825C7DE02B1&smsid=xxxxxxxx
```

Parameter	Type	Description
sender	string	The message sender
smsid	string	The unique id that the message has.
dlrDate	string (YYYY-MM-DD HH:MM:SS)	The date that the message delivered
downloadUrl	string	The URL of the file to be downloaded.

If additional callback data are predefined, then they will be appended to the query.

1.4.4 User Subscribe / Unsubscribe Callback

When a user unsubscribes from the sender subscription lists, a response can be invoked to a predefined URL.

```
https://www.example.com/example?receiver=3069XXXXXXXX&dlrDate=2021-01-8
10:13:21&sender=Test%202way&status=5&statusDescription=unsubscribed
```

Parameter	Type	Description
sender	string	The message sender
receiver	string	The destination phone number
status	int	The status number (10: UnSubscribed / 9: Subscribed)
statusDescription	string	The status description (UnSubscribed / Subscribed)
dlrDate	string (YYYY-MM-DD HH:MM:SS)	The date that the callback was send

If additional callback data are predefined, then they will be appended to the query.

For the time being this service is provided only for **Viber** subscribed Users.

2. DLR Method

Description

Using this method, you can retrieve information on sent text messages and check their status in real-time.

URL to web service operation

```
https://services.yuboto.com/omni/v1/Dlr
```

2.1 Method Parameters, Request, and Response

Request Parameters

The variables used to retrieve information on sent text messages and check their status in real-time are:

Variables	Description	Permitted Values	Required
id	The id of message status.	String	Yes

Request Example (POST)

```
{
    "id" : "54E3B5F5-2CF3-412E-80A6-A324D94500F6"
}
```

Request Example (GET)

1- Header Authentication:

```
https://services.yuboto.com/omni/v1/Dlr?id=54E3B5F5-2CF3-412E-80A6-
A324D94500F6
```

2- URL Authentication (apiKey Parameter Authentication Example):

```
https://services.yuboto.com/omni/v1/Dlr?id=54E3B5F5-2CF3-412E-80A6-
A324D94500F6&apiKey=XFv4RERCQTktNjBEQi00ODAyLUE3NTEtMTUyMDkzMsU4QkdFL
```

Response Example

```
{
    "ErrorCode": 0,
    "ErrorMessage": "",
    "id": "54E3B5F5-2CF3-412E-80A6-A324D94500F6",
    "phonenumber": "306936XXXXXXX",
    "option1": "option1 value",
    "option2": "option2 value",
    "dlr":[
        {
            "channel": "viber",
            "priority": 0,
            "status": "Not Delivered",
            "cost": 1,
            "sender": "Test 2way",
            "text": "This is a demo viber msg",
            "submitDate": "\/Date(1500550221991)\/",
            "dlrDate": "\/Date(1500550221990)\/"
        },
        {
            "channel": "sms",
            "priority": 1,
            "status": "Delivered",
            "cost": 1,
            "sender": "TestSender",
            "text": "This is a demo sms msg",
            "submitDate": "\/Date(1500550281991)\/",
            "dlrDate": "\/Date(1500550341991)\/"
        }
    ]
}
```

ErrorCode: The response error code for this call. This will be 0 if successful.

ErrorMessage: The response error message. This will be null if successful.

id: The ID of message status.

phonenumber: Refers to the phone number of the recipient or recipients of the text message.

option1: The value included in the call to the provided callback_url.

option2: The value included in the call to the provided callback_url.

dlr: A list with DLR channels and their details.

- **channel:** The message channel related to DLR request. Possible values are: Viber or SMS.
- **priority:** Indicates which channel has priority when it comes to OMNI messaging (default value is: 0).
- **status:** The status that the message has.
- **cost:** The cost of the message.
- **sender:** The sender of the message.
- **text:** The text that the message has.
- **submitDate:** The date the message was sent.
- **dlrDate:** The date the message was delivered.

If this method called successfully then the ErrorCode has the value 0 and the ErrorMessage contains a zero-length string. If an error occurred then, consult the error message that appears.

2.2 Status of Messages

The following table shows the possible status of a message (SMS or Viber):

Status	Viber	SMS	Description	Final Status
Sent	Yes	Yes	The message has been sent to the final network to be delivered	No
Pending	Yes	Yes	The state of the message is not yet known	No
Submitted	Yes	Yes	The message has been routed for sending	No
Scheduled	Yes	Yes	The message has been scheduled for sending in a future time	No
Fallbacksms	Yes	No	The message wasn't delivered via Viber and it has been forwarded for sending via SMS	No
Not Delivered	Yes	Yes	The message was not delivered********	Yes
Unknown	Yes	No	The message took another status from the ones we have available *****	Yes
Error	Yes	Yes	Error during the sending process	Yes
Expired	Yes	Yes	The message wasn't delivered within the time frame we've set *****	Yes
Failed	Yes	Yes	Sending failed due to insufficient balance in your account ***	Yes
Rejected	Yes	Yes	A message is rejected when the system rejects it's sending due to routing reasons. Wrong number or not supported termination to a specific network are common reasons *********	Yes
Canceled	Yes	Yes	The sending of the message has been canceled	Yes
Seen	Yes	Yes	The recipient has seen the message	Yes
Clicked	Yes	Yes	The recipient has clicked a link in the message	Yes
Blacklisted	Yes	Yes	The recipient's number is blacklisted	Yes
Unsubscribed	Yes	Yes	The recipient clicked on the unsubscribed option	Yes
Blocked	Yes	No	The Viber Sender Id that sent the message has been blocked	Yes

** Indicates if this is the final status of the message or it is going to change.*

*** Some of the possible reasons for the failure of delivery might be:*

i) Invalid telephone number

ii) Telephone deactivated or switched off. In the last two cases, the SMSC holds the message for 3 days and before rejecting it, allows you to select a shorter time period (the variable validity of SmsObj see par. 2.2)

**** Delivery fails when there are no available Credits to your account.*

***** Messages are rejected when the recipient of the SMS or Viber message has an invalid format or when your account or Yuboto platform does not support it.*

Failed or Rejected messages are not charged (to your account).

***** *These statuses (final status) cause a fallback to your 2nd priority channel. For example, if your priority channel is Viber, at first Yuboto will try to deliver a Viber service message. If the Viber Service Message is undeliverable for whatsoever reason, Yuboto will send an SMS message.*

Possible reasons for Viber Service Message non-delivery are:

- Subscriber does not have the Viber app installed on his/her device.
- Subscriber is not reachable within given TTL.
- Subscriber has Viber app that does not support Viber Service Messages (e.g., Windows Phone OS version of Viber app).

3. Cost Method

Description

Through the following method, you can request the cost of sending a simple SMS or Viber.

URL to web service operation

```
https://services.yuboto.com/omni/v1/Cost
```

Request Parameters

The variables used to request the cost of sending a simple SMS to one or multiple recipients are:

Variables	Description	Permitted Values	Required
iso2	The ISO_3166-1_alpha-2 code of the country.	String. 2-letter code	Yes*
phonenumber	Refers to the phone number of the recipient of the text message.	String	Yes*
channel	The channel that the message will be sent (SMS or Viber).	String	Yes**

** One of two of these parameters must always exist.*
** In case you have an **OMNI API Key** (you send SMS and/or Viber messages), you need to specify for which channel you want to learn the cost. If your channel is not OMNI, then this parameter is not required.

Request Example (POST)

```
{
    "iso2" : "gr",
    "channel" : "sms"
}
```

Request Example (GET)

1- Header Authentication:

```
https://services.yuboto.com/omni/v1/Cost?iso2=gr&channel=sms
```

2- URL Authentication (apiKey Parameter Authentication Example):

```
https://services.yuboto.com/omni/v1/Cost?iso2=gr&channel=sms&apiKey=XFv4RERCQ
TktNjBEQi00ODAyLUE3NTEtMTUyMDkzMsU4QkdFL
```

Response Example

```
{
    "ErrorCode": 0,
    "ErrorMessage": "",
    "channel": "sms",
    "type": "credits",
    "costInfo":[
        {
            "networkName": "Network name1",
            "cost": 1
        },
        {
            "networkName": "Network name2",
            "cost": 1
        }
    ]
}
```

ErrorCode: The response error code for this call. This will be 0 if successful.

ErrorMessage: The response error message. This will be null if successful.

channel: The channel that the message will be sent (SMS or Viber).

type: Indicates the type of your cost (e.g. credits or money).

costInfo: A list with all the details about the cost of sending a simple SMS or Viber message to one or multiple recipients.

- **networkName:** The name of the network (e.g. "VODAFONE - PANAFON Hellenic Telecommunications Company").
- **cost:** The cost of sending a simple SMS or Viber message to one or multiple recipients.

If this method called successfully then the ErrorCode has the value 0 and the ErrorMessage contains a zero-length string. If an error occurred then, consult the error message that appears.

4. Cost Details Method

Description

Using this method, you can retrieve the cost details of sending a simple SMS or Viber message for a specific iso2. The difference from the Cost Method is that this method returns for a specific iso2, the mcc and mnc.

URL to web service operation

```
https://services.yuboto.com/omni/v1/CostDetails
```

Request Parameters

The variables used to retrieve the cost details of sending a simple SMS or Viber message to one or multiple recipients are:

Variables	Description	Permitted Values	Required

iso2	The ISO_3166-1_alpha-2 code of the country.	String. 2-letter code	Yes
channel	The channel that the message will be sent (SMS or Viber).	String	Yes*

* In case you have an **OMNI API Key** (you send SMS and/or Viber messages), you need to specify for which channel you want to learn the cost. If your API Key is not OMNI, then this parameter is not required.

Request Example (POST)

```
{
    "iso2" : "gr",
    "channel" : "sms"
}
```

Request Example (GET)

1- Header Authentication:

```
https://services.yuboto.com/omni/v1/CostDetails?iso2=gr
```

2- URL Authentication (apiKey Parameter Authentication Example):

```
https://services.yuboto.com/omni/v1/CostDetails?iso2=gr&apiKey=XFv4RERCQTktNj
BEQi00ODAyLUE3NTEtMTUyMDkzMsU4QkdFL
```

Response Example

```
{
    "costInfoDetails":[
        {
            "networkName": "Wind Hellas Telecommunications SA",
            "mcc": "202",
            "mnc": "009",
            "cost": 1.00
        },
        {
            "networkName": "Wind Hellas Telecommunications SA",
            "mcc": "202",
            "mnc": "010",
            "cost": 1.00
        },
        {
            "networkName": "VODAFONE - PANAFON Hellenic Telecommunications
Company SA",
            "mcc": "202",
            "mnc": "005",
            "cost": 1.00
        },
        {
            "networkName": "COSMOTE Mobile Telecommunications SA",
            "mcc": "202",
            "mnc": "001",
            "cost": 1.00
        },
        {
            "networkName": "COSMOTE Mobile Telecommunications SA",
            "mcc": "202",
            "mnc": "002",
            "cost": 1.00
        }
    ],
    "ErrorCode": 0,
    "ErrorMessage": null,
    "channel": "sms",
    "type": "credits"
}
```

ErrorCode: The response error code for this call. This will be 0 if successful.

ErrorMessage: The response error message. This will be null if successful.

channel: The channel that the message will be sent (SMS or Viber).

type: Indicates the type of your cost (e.g. credits or money).

costInfoDetails: A list with all the details about the cost info of sending a simple SMS or Viber message to one or multiple recipients.

- **networkName:** The name of the network (e.g. “VODAFONE - PANAFON Hellenic Telecommunications Company”).
- **mcc:** The mobile country code (MCC) consists of 3 decimal digits (e.g. “202”).
- **mnc:** The mobile network code (MNC) consists of 2 or 3 decimal digits (for example MNC of 001 is not the same as MNC of 01). The first digit of the mobile country code identifies the geographic region as follows (the digits 1 and 8 are not used):
    - 0 - Test networks
    - 2 - Europe
    - 3 - North America and the Caribbean
    - 4 - Asia and the Middle East
    - 5 - Oceania
    - 6 - Africa
    - 7 - South and Central America

    - 9 - Worldwide (Satellite, Air - aboard aircraft, Maritime - aboard ships, Antarctica)
- **cost:** The cost of sending a simple SMS or Viber message to one or multiple recipients.

If this method called successfully then the ErrorCode has the value 0 and the ErrorMessage contains a zero-length string. If an error occurred then, consult the error message that appears.

5. User Balance Method

Description

Through the following method, you can retrieve information on your current balance.

URL to web service operation

```
https://services.yuboto.com/omni/v1/UserBalance
```

Request Example (POST)

No required parameters for this method.

Request Example (GET)

1- Header Authentication:

```
https://services.yuboto.com/omni/v1/UserBalance
```

2- URL Authentication (apiKey Parameter Authentication Example):

```
https://services.yuboto.com/omni/v1/UserBalance?apiKey=XFv4RERCQTktNjBEQi00OD
AyLUE3NTEtMTUyMDkzMsU4QkdFL
```

Response Example

```
{
    "ErrorCode": 0,
    "ErrorMessage": "",
    "currentBalance": 110,
    "actualBalance": 10,
    "limitBalance": 100,
    "type": "credits"
}
```

ErrorCode: The response error code for this call. This will be 0 if successful.

ErrorMessage: The response error message. This will be null if successful.

currentBalance: Your current balance in credits or money, including the limitBalance.

actualBalance: The actual balance in credits or money.

limitBalance: This is your account's overdraft. To what limit your account can send messaging (default value is: 0).

type: The type of your balance based on the user's configuration (e.g. credits or money).

If the method called successfully then the ErrorCode has the value 0 and the ErrorMessage contains a zero-length string. If an error occurred then, consult the error message that appears.

6. User Subscription Method

Description

UserSubscription method returns active subscription information per channel of the user.

URL to web service operation

```
https://services.yuboto.com/omni/v1/UserSubscription
```

Request Example (POST)

No required parameters for this method.

Request Example (GET)

1- Header Authentication:

```
https://services.yuboto.com/omni/v1/UserSubscription
```

2- URL Authentication (apiKey Parameter Authentication Example):

```
https://services.yuboto.com/omni/v1/UserSubscription?apiKey=XFv4RERCQTktNjBEQ
i00ODAyLUE3NTEtMTUyMDkzMsU4QkdFL
```

Response Example

```
{
    "ErrorCode": 0,
    "ErrorMessage": null,
    "type": "money",
    "userSubscriptions":
    [
        {
            "initialBalance": 0.2000,
            "currentBalance": 0.2000,
            "actualBalance": 0.2000,
            "limitBalance": 0.0000,
            "dateStart": "\/Date(1522161543257)\/",
            "dateEnd": "\/Date(1590958799000)\/",
            "subscriptionChannels":[
                {
                    "channel":"VIBER",
                    "senderName":"Viber Sender ID"
                }
            ]
        },
        {
            "initialBalance": 400.0000,
            "currentBalance": 200.0000,
            "actualBalance": 210.0000,
            "limitBalance": -10.0000,
            "dateStart": "\/Date(1538653127000)\/",
            "dateEnd": "\/Date(1601897927000)\/",
            "subscriptionChannels":[
                {
                    "channel": "SMS",
                    "senderName": "Any"
                }
            ]
        }
    ]
}
```

ErrorCode: The response error code for this call. This will be 0 if successful.

ErrorMessage: The response error message. This will be null if successful.

type: The type of subscription.

userSubscritpions: The list of subscriptions and channels.

- **initialBalance :** The initial balance of the subscription.
- **currentBalance:** The current balance of the subscription.
- **actualBalance:** The current balance minus overdraft.
- **limitBalance:** Overdraft value if any.
- **dateStart:** The initial starting date of the subscription.
- **dateEnd:** The expiration date of the subscription.
- **subscriptionChannels:** The list of subscription channels.
    - **channel:** The channel of subscription.
    - **senderName:** The sender Name of subscription channel.

If the method called successfully then the ErrorCode has the value 0 and the ErrorMessage contains a zero-length string. If an error occurred then, consult the error message that appears.

For the cases of Master – Childs accounts all channel info appears to the master account repeated for each child accounts are using the sender id.

Example

```
{
    "ErrorCode": 0,
    "ErrorMessage": null,
    "type": "money",
    "userSubscriptions":
    [
      {
            "initialBalance": 0.2000,
            "currentBalance": 0.2000,
            "actualBalance": 0.2000,
            "limitBalance": 0.0000,
            "dateStart": "\/Date(1522161543257)\/",
            "dateEnd": "\/Date(1590958799000)\/",
            "subscriptionChannels":[
                {
                    "channel":"VIBER",
                    "senderName":"Viber Sender ID"
                }
            ]
        },
        {
            "initialBalance": 400.0000,
            "currentBalance": 200.0000,
            "actualBalance": 210.0000,
            "limitBalance": -10.0000,
            "dateStart": "\/Date(1538653127000)\/",
            "dateEnd": "\/Date(1601897927000)\/",
            "subscriptionChannels":[
                {
                    "channel": "SMS",
                    "senderName": "Any"
                }
            ]
        },
        {
            "initialBalance": 300.0000,
            "currentBalance": 300.0000,
            "actualBalance": 300.0000,
            "limitBalance":.0000,
            "dateStart": "\/Date(1538653380000)\/",
            "dateEnd": "\/Date(1601897927000)\/",
            "subscriptionChannels":[
                {
                    "channel": "VIBER",
                    "senderName": "Sender Name"
                }
            ]
        },
        {
            "initialBalance": 300.0000,
            "currentBalance": 300.0000,
            "actualBalance": 300.0000,
            "limitBalance":.0000,
            "dateStart": "\/Date(1538653380000)\/",
            "dateEnd": "\/Date(1601897927000)\/",
            "subscriptionChannels":[
                {
                    "channel": "VIBER",
                    "senderName": "Some Name"
                },
                {
                    "channel": "VIBER",
                    "senderName": "Other Name"
                }
            ]
```

```
                }
            ]
}
```

7. Cancel Method

Description

Through the following method, you can cancel a scheduled message before the scheduled date and time. You can cancel sending a message up to **three minutes** before the time it is scheduled to send.

URL to web service operation

```
https://services.yuboto.com/omni/v1/Cancel
```

Request Parameters

The variables used to create a user are:

Variables	Description	Permitted Values	Required
id	The id of message status.	String	Yes

Request Example (POST)

```
{
      "id" : "601756D5-6537-4DC5-BD07-10D95BF1621E"
}
```

Request Example (GET)

1- Header Authentication:

```
https://services.yuboto.com/omni/v1/Cancel?id=601756D5-6537-4DC5-BD07-
10D95BF1621E
```

2- URL Authentication (apiKey Parameter Authentication Example):

```
https://services.yuboto.com/omni/v1/Cancel?id=601756D5-6537-4DC5-BD07-
10D95BF1621E&apiKey=XFv4RERCQTktNjBEQi00ODAyLUE3NTEtMTUyMDkzMsU4QkdFL
```

Response Example

```
{
     "ErrorCode":0,
     "ErrorMessage":"",
     "channel":"sms",
     "id":"601756D5-6537-4DC5-BD07-10D95BF1621E",
     "status":"Canceled"
}
```

ErrorCode: The response error code for this call. This will be 0 if successful.

ErrorMessage: The response error message. This will be null if successful.

channel: The channel that the message is scheduled to be sent (SMS or Viber).

id: The ID of message status.

status: The status of the message.

If the method called successfully then the ErrorCode has the value 0 and the ErrorMessage contains a zero-length string. If an error occurred then, consult the error message that appears.

8. Create Key Method

Description

This method creates an API Key for your subaccounts. Contact with Account Manager, in order to give you more information about how to get an API Key. Thanks to this method, you can provide your subaccounts with an API Key that they can use.

URL to web service operation

```
https://services.yuboto.com/omni/v1/CreateKey
```

Request Parameters

The variables used to create a user's API Key are:

Variables	Description	Permitted Values	Required
username	The username of your subaccount.	String	Yes

Request Example (POST)

```
{
    "username" : "demouser22"
}
```

Request Example (GET)

1- Header Authentication:

```
https://services.yuboto.com/omni/v1/CreateKey?username=demouser22
```

2- URL Authentication (apiKey Parameter Authentication Example):

```
https://services.yuboto.com/omni/v1/CreateKey?username=demouser22&apiKey=XFv4
RERCQTktNjBEQi00ODAyLUE3NTEtMTUyMDkzMsU4QkdFL
```

Response Example

```
{
    "ErrorCode": 0,
    "ErrorMessage": "",
    "username": "demouser22",
    "apiKey": "-----NewApiKey-----",
    "channel": "sms"
}
```

ErrorCode: The response error code for this call. This will be 0 if successful.

ErrorMessage: The response error message. This will be null if successful.

username: The user's username.

apiKey: The unique API Key of the user.

channel: The channel that your API Key can be used. Possible values are:

- **viber** - Sending only VIBER message.
- **sms** - Sending only SMS message.
- **omni** - A combination of all available channels. In case there are more than two channels, then the system will see the priority of each channel and send the messages to the first priority channel.

If the method called successfully then the ErrorCode has the value 0 and the ErrorMessage contains a zero-length string. If an error occurred then, consult the error message that appears.

9. Two-Factor Authentication Validation Method/OTP

Description

This method validates the pin for a specific smsid for two-factor authentication messages.

URL to web service operation

```
https://services.yuboto.com/omni/v1/verifypin
```

Request Parameters

The variables used to create a user’s API Key are:

Variables	Description	Permitted Values	Required
id	Sms id system returned upon submission.	String	Yes
pin	User entered pin for validation.	String	Yes

Request Example (POST)

```
{
    "id" : "A25896DF-6219-1CRN-U59M-6865BO9JQ561",
    "pin" : "5327"
}
```

Request Example (GET)

1- Header Authentication:

```
https://services.yuboto.com/omni/v1/verifypin?id=A25896DF-6219-1CRN-U59M-
6865BO9JQ561&pin=5327
```

2- URL Authentication (apiKey Parameter Authentication Example):

```
https://services.yuboto.com/omni/v1/verifypin?id=A25896DF-6219-1CRN-U59M-
6865BO9JQ561&pin=5327&apiKey=XFv4RERCQTktNjBEQi00ODAyLUE3NTEtMTUyMDkzMsU4QkdF
L
```

Response Example

```
{
    "ErrorCode":0,
    "ErrorMessage":"",
    "id":"---sms id---",
    "pin":"--pin---",
    "phonenumber":"—receiver’s phonenumber ---",
    "status ":"valid",
}
```

If validation fails then the system will return error codes:

Error Code	Description
37	Invalid PIN.
38	PIN Expired.

10. Phone Number Reporting

Description

This method retrieves all existing send/delivery reports for previous campaigns based on the recipient's phone number.

URL to web service operation

```
https://services.yuboto.com/omni/v1/DlrPhonenumber
```

Request Parameters

The variables used to retrieve existing reports for specific recipient:

Variables	Description	Permitted Values	Required
phonenumber	Recipient's Phone number in international format.	String	Yes

Request Example (POST)

```
{
    "phonenumber" : "3069XXXXXXXX"
}
```

Request Example (GET)

1- Header Authentication:

```
https://services.yuboto.com/omni/v1/DlrPhonenumber?phonenumber=3069XXXXXXXX
```

2- URL Authentication (apiKey Parameter Authentication Example):

```
https://services.yuboto.com/omni/v1/DlrPhonenumber?phonenumber=3069XXXXXXXX&a
piKey=XFv4RERCQTktNjBEQi00ODAyLUE3NTEtMTUyMDkzMsU4QkdFL
```

Response Example

```
{
    "ErrorCode": 0,
    "ErrorMessage": null,
    "phonenumber": "3069XXXXXXXX",
    "dlr": [
        {
            "id": "XXXX-XXXX-XXXX-XXXX-XXXXX",
            "system": "Api",
            "channel": "sms",
            "priority": 0,
            "status": "Delivered",
            "cost": 0.04,
            "sender": "Test",
            "text": "This is a test",
            "submitDate": "2019-01-11 18:38:03",
            "dlrDate": "2019-01-11 18:38:09"
        }
    ]
}
```

ErrorCode: The response error code for this call. This will be 0 if successful.

ErrorMessage: The response error message. This will be null if successful.

phoneumber: The user's username.

dlr: A list with DLR channels and their details.

- **id:** The id of the message.
- **system:** The system used.
- **channel:** The channel used. Viber / SMS.
- **priority:** Indicates which channel has priority when it comes to OMNI messaging (default value is: 0).
- **status:** The status that the message has.
- **cost:** The cost of the message.
- **sender:** The sender of the message.
- **text:** The text that the message has.
- **submitDate:** The date the message was sent.
- **dlrDate:** The date the message was delivered.

If the method called successfully then the ErrorCode has the value 0 and the ErrorMessage contains a zero-length string. If an error occurred then, consult the error message that appears.

11. Subscriber Lists

11.1 Get Subscriber Lists

Description

GetSubscriberLists method retrieves all existing Subscriber Lists.

URL to web service operation

```
https://services.yuboto.com/omni/v1/GetSubscriberLists
```

Request Example (POST)

No required parameters for this method.

Request Example (GET)

1- Header Authentication:

```
 https://services.yuboto.com/omni/v1/GetSubscriberLists
```

2- URL Authentication (apiKey Parameter Authentication Example):

```
https://services.yuboto.com/omni/v1/GetSubscriberLists?apiKey=XFv4RERCQTktNjB
EQi00ODAyLUE3NTEtMTUyMDkzMsU4QkdFL
```

Response Example

```
{
    "ErrorCode": 0,
    "ErrorMessage": null,
    "Subscriberlists": [
        {
           "id": "xxxxxxxx-xxxx-xxxx"
           ,"name": "ListName"
           ,"activeContacts": 2
           ,"totalContacts": 2
           }
     ]
}
```

ErrorCode: The response error code for this call. This will be 0 if successful.

ErrorMessage: The response error message. This will be null if successful.

SubscriberLists: A list with all subscriber lists.

- **id:** the id of each subscriber list.

- **name:** The name of each subscriber list.
- **activeContacts:** The number of active contacts in the list.
- **totalContacts:** The total number of contacts in the list (active & inactive).

If the method called successfully then the ErrorCode has the value 0 and the ErrorMessage contains a zero-length string. If an error occurred then, consult the error message that appears.

11.2 Create Subscriber List

Description

CreateSubscriberList method creates a new Subscriber List.

URL to web service operation

```
https://services.yuboto.com/omni/v1/CreateSubscriberList
```

Request Parameters

Variables	Description	Permitted Values	Required
name	Refers to the name of the subscriber list.	String	Yes

Request Example (POST)

```
{
    "name" : "SubscriberListName"
}
```

Request Example (GET)

1- Header Authentication:

```
https://services.yuboto.com/omni/v1/CreateSubscriberList?name=SubscriberListN
ame
```

2- URL Authentication (apiKey Parameter Authentication Example):

```
https://services.yuboto.com/omni/v1/CreateSubscriberList?name=SubscriberListN
ame&apiKey=XFv4RERCQTktNjBEQi00ODAyLUE3NTEtMTUyMDkzMsU4QkdFL
```

Response Example

```
{
    "ErrorCode": 0,
    "ErrorMessage": null,
    "Status": "created",
    "key": "xxxxxxxx-xxxx-xxxx"
}
```

ErrorCode: The response error code for this call. This will be 0 if successful.

ErrorMessage: The response error message. This will be null if successful.

Status: The performed action status.

key: The unique id of the newly created record.

If the method called successfully then the ErrorCode has the value 0 and the ErrorMessage contains a zero-length string. If an error occurred then, consult the error message that appears.

11.3 Edit Subscriber List

Description

EditSubscriberList method updates an existing Subscriber List.

URL to web service operation

```
https://services.yuboto.com/omni/v1/EditSubscriberList
```

Request Parameters

Variables	Description	Permitted Values	Required
id	Refers to the Unique ID of the subscriber list.	String	Yes
name	Refers to the name of the subscriber list (to be updated).	String	Yes

Request Example (POST)

```
{
    "id" : "xxxxxxxx-xxxx-xxxx",
    "name" : "NewSubscriberListName"
}
```

Request Example (GET)

1- Header Authentication:

```
https://services.yuboto.com/omni/v1/EditSubscriberList?id=xxxxxxxx-xxxx-xxxx
&name=NewSubscriberListName
```

2- URL Authentication (apiKey Parameter Authentication Example):

```
https://services.yuboto.com/omni/v1/EditSubscriberList?id=xxxxxxxx-xxxx-
xxxx&name=NewSubscriberListName&apiKey=XFv4RERCQTktNjBEQi00ODAyLUE3NTEtMTUyMD
kzMsU4QkdFL
```

Response Example

```
{
    "ErrorCode": 0,
    "ErrorMessage": null,
    "Status": "updated",
    "key": "xxxxxxxx-xxxx-xxxx"
}
```

ErrorCode: The response error code for this call. This will be 0 if successful.

ErrorMessage: The response error message. This will be null if successful.

Status: The performed action status.

key: The unique ID of the updated record.

If the method called successfully then the ErrorCode has the value 0 and the ErrorMessage contains a zero-length string. If an error occurred then, consult the error message that appears.

11.4 Delete Subscriber List

Description

DeleteSubscriberList method deletes an existing Subscriber List.

URL to web service operation

```
https://services.yuboto.com/omni/v1/DeleteSubscriberList
```

Request Parameters

Variables	Description	Permitted Values	Required

id	Refers to the Unique ID of the subscriber list.	String	Yes

Request Example (POST)

```
{
    "id" : "xxxxxxxx-xxxx-xxxx"
}
```

Request Example (GET)

1- Header Authentication:

```
https://services.yuboto.com/omni/v1/DeleteSubscriberList?id=xxxxxxxx-xxxx-
xxxx
```

2- URL Authentication (apiKey Parameter Authentication Example):

```
https://services.yuboto.com/omni/v1/DeleteSubscriberList?id=xxxxxxxx-xxxx-
xxxx&apiKey=XFv4RERCQTktNjBEQi00ODAyLUE3NTEtMTUyMDkzMsU4QkdFL
```

Response Example

```
{
    "ErrorCode": 0,
    "ErrorMessage": null,
    "Status": "deleted",
    "key": "xxxxxxxx-xxxx-xxxx"
}
```

ErrorCode: The response error code for this call. This will be 0 if successful.

ErrorMessage: The response error message. This will be null if successful.

Status: The performed action status.

key: The unique ID of the deleted record.

If the method called successfully then the ErrorCode has the value 0 and the ErrorMessage contains a zero-length string. If an error occurred then, consult the error message that appears.

12. Contacts

12.1 Get Contacts

Description

GetContacts method retrieves the Contacts belonging to a specified list. The response includes paged results and the total records count.

URL to web service operation

```
https://services.yuboto.com/omni/v1/GetContacts
```

Request Parameters

Variables	Description	Permitted Values	Required
subscriberlistid	Refers to the subscriber list unique ID.	String	Yes
pagenumber	Indicates the page number of the available data.	Integer.	Yes
pagesize	Indicates the page record size of the response. The maximum page size is 1000.	Integer. Number between 1 and 1000	Yes

Request Example (POST)

```
{
   "subscriberlistid": "xxxxxxxx-xxxx-xxxx",
   "pagenumber": 1,
   "pagesize": 100
}
```

Request Example (GET)

1- Header Authentication:

```
https://services.yuboto.com/omni/v1/GetContacts?subscriberlistid=xxxxxxxx-
xxxx-xxxx&pagenumber=1&pagesize=100
```

2- URL Authentication (apiKey Parameter Authentication Example):

```
https://services.yuboto.com/omni/v1/GetContacts?subscriberlistid=xxxxxxxx-
xxxx-
xxxx&pagenumber=1&pagesize=100&apiKey=XFv4RERCQTktNjBEQi00ODAyLUE3NTEtMTUyMDk
zMsU4QkdFL
```

Response Example

```
{
    "ErrorCode": 0,
    "ErrorMessage": null,
    "TotalRecords": 1,
    "contacts": [
        {
            "id": "xxxxxxxx-xxxx-xxxx",
            "subscriberlistid": "xxxxxxxx-xxxx-xxxx",
            "phonenumber": "3069XXXXXXXX",
            "name": "string",
            "surname": "string",
            "address": "string",
            "title": "string",
            "option1": "string",
            "option2": "string",
            "active": "true",
            "customfields": [
                {
                    "name": "fieldname",
                    "value": "fieldvalue"
                }
            ]
        }
    ]
}
```

ErrorCode: The response error code for this call. This will be 0 if successful.

ErrorMessage: The response error message. This will be null if successful.

TotalRecords: The total contact count of the subscriber list.

contacts: A pages list with the contact information.

- **id:** The unique ID of the contact.
- **subscriberlistid:** The unique ID of the subscriber list.
- **phonenumber:** The channel used. Viber / SMS.
- **name:** Contact name.
- **surname:** Contact surname.
- **address:** Contact addresses.
- **title:** Contact title.
- **option1:** Optional free text 1.
- **option2:** Optional free text 2.
- **active:** The status of the contact (true / false).
- **customfields:** A list of key/value pair: name / value of the custom fields.

If the method called successfully then the ErrorCode has the value 0 and the ErrorMessage contains a zero-length string. If an error occurred then, consult the error message that appears.

12.2 Search Contacts

Description

SearchContacts method retrieves the Contacts belonging to a specified set of lists, using filters. The response includes paged results and the total records count.

URL to web service operation

```
https://services.yuboto.com/omni/v1/SearchContacts
```

Request Parameters

Variables	Description	Permitted Values	Required
subscriberlists	An array of the subscriber list unique ids to search.	String Minimum in Unique id	Yes
pagenumber	Indicates the page number of the available data.	Integer.	Yes
pagesize	Indicates the page record size of the response. The maximum page size is 1000.	Integer. Number between 1 and 1000	Yes
Filters	An array of conditions for filtering the search. Available condition keys : contain,startswith,endswith,greater,smaller,greaterorequal,notequal,notcontains	Format of each filter { field: fieldname, condition : conditionName, value : valueToCompaire }	No

Request Example (POST)

```
{
    "subscriberlists": ["xxxxxxxx-xxxx-xxxx", "xxxxxxxx-xxxx-xxxx"],
    "pagenumber": 1,
    "pagesize": 100,
    "filters": [
        {
            "field": "firstname",
            "condition": "contains",
            "value": "anystring"
        }
    ]
}
```

Request Example (GET)

1- Header Authentication:

```
https://services.yuboto.com/omni/v1/SearchContacts?subscriberlists[]=xxxxxxxx
-xxxx-xxx0&subscriberlists[]=xxxxxxxx-xxxx-
xxx1&pagenumber=1&pagesize=100&filters[0][field]=firstname&filters[0][conditi
on]=contains&filters[0][value]=anystring
```

2- URL Authentication (apiKey Parameter Authentication Example):

```
https://services.yuboto.com/omni/v1/SearchContacts?subscriberlists[]=xxxxxxxx
-xxxx-xxx0&subscriberlists[]=xxxxxxxx-xxxx-
xxx1&pagenumber=1&pagesize=100&filters[0][field]=firstname&filters[0][conditi
on]=contains&filters[0][value]=anystring&pagesize=100&apiKey=XFv4RERCQTktNjBE
Qi00ODAyLUE3NTEtMTUyMDkzMsU4QkdFL
```

Response Example

```
{
    "ErrorCode": 0,
    "ErrorMessage": null,
    "TotalRecords": 1,
    "contacts": [
        {
            "id": "xxxxxxxx-xxxx-xxxx",
            "subscriberlistid": "xxxxxxxx-xxxx-xxxx",
            "phonenumber": "3069XXXXXXXX",
            "name": "string",
            "surname": "string",
            "address": "string",
            "title": "string",
            "option1": "string",
            "option2": "string",
            "active": "true",
            "customfields": [
                {
                    "name": "fieldname",
                    "value": "fieldvalue"
                }
            ]
        }
    ]
}
```

ErrorCode: The response error code for this call. This will be 0 if successful.

ErrorMessage: The response error message. This will be null if successful.

TotalRecords: The total contact count of the subscriber list.

contacts: A pages list with the contact information.

- **id:** The unique ID of the contact.
- **subscriberlistid:** The unique id of the subscriber list.
- **phonenumber:** The channel used. Viber / SMS.
- **name:** Contact name.
- **surname:** Contact surname.
- **address:** Contact addresses.
- **title:** Contact title.
- **option1:** Optional free text 1.
- **option2:** Optional free text 2.
- **active:** The status of the contact (true / false).
- **customfields:** A list of key/value pair: name / value of the custom fields.

If the method called successfully then the ErrorCode has the value 0 and the ErrorMessage contains a zero-length string. If an error occurred then, consult the error message that appears.

12.3 Create Contact

Description

CreateContact method creates a new Contact in the defined Contact list.

URL to web service operation

```
https://services.yuboto.com/omni/v1/CreateContact
```

Request Parameters

Variables	Description	Permitted Values	Required
subscriberlistid	Refers to the subscriber list unique ID.	String	Yes
phonenumber	Refers to the phone number of the recipient or recipients of the text message (use array for multiple recipients).	String/Array. Use country code without + or 00.	Yes
name	Optional field of any string.	String	Yes
surname	Optional field of any string.	String	No
title	Optional field of any string.	String	No
address	Optional field of any string.	String	No
email	Optional field of any string.	String	No
active	Refers to contact status.	String “true” ,“false”	Yes
Option1	Optional field of any string.	String	No

Option2	Optional field of any string.	String	No
customfields	Key value pair of custom field names and their respective values. Key-value should match the custom fields defined in Octapush.	CustomField {"FieldName":"FieldValue", "FieldName2":"FieldValue2" ….}	No

Request Example (POST)

```
{
    "subscriberlistid": "xxxxxxxx-xxxx-xxxx",
    "phonenumber": "3069XXXXXXXX",
    "name": "ContactName",
    "surname": "ContactSurname",
    "title": "Mr",
    "address": "olympias 2",
    "email": "info@yuboto.com",
    "active": "true",
    "customfields": {"FieldName":"FieldValue"}
}
```

Request Example (GET)

1- Header Authentication:

```
https://services.yuboto.com/omni/v1/CreateContact?subscriberlistid= xxxxxxxx-
xxxx-
xxxx&phonenumber=3069XXXXXXXX&name=ContactName&surname=ContactSurname&title=M
r&address=olympias2&email=info@yuboto.com&active=true&customfields[FieldName]
=FieldValue
```

2- URL Authentication (apiKey Parameter Authentication Example):

```
https://services.yuboto.com/omni/v1/CreateContact?subscriberlistid= xxxxxxxx-
xxxx-
xxxx&phonenumber=3069XXXXXXXX&name=ContactName&surname=ContactSurname&title=M
r&address=olympias2&email=info@yuboto.com&active=true&customfields[FieldName]
=FieldValue&apiKey=XFv4RERCQTktNjBEQi00ODAyLUE3NTEtMTUyMDkzMsU4QkdFL
```

Response Example

```
{
    "ErrorCode": 0,
    "ErrorMessage": null,
    "Status": "created",
    "key": "xxxxxxxx-xxxx-xxxx"
}
```

ErrorCode: The response error code for this call. This will be 0 if successful.

ErrorMessage: The response error message. This will be null if successful.

Status: The performed action status.

key: The unique ID of the newly created record.

If the method called successfully then the ErrorCode has the value 0 and the ErrorMessage contains a zero-length string. If an error occurred then, consult the error message that appears.

12.4 Edit Contact

Description

EditContact method updates an existing Contact in the defined Contact list.

URL to web service operation

```
https://services.yuboto.com/omni/v1/EditContact
```

Request Parameters

z	Description	Permitted Values	Required
id	Refers to the contact unique ID to be updated.	String	Yes
subscriberlistid	Refers to the subscriber list unique ID.	String	Yes
phonenumber	Refers to the phone number of the recipient or recipients of the text message (use array for multiple recipients).	String/Array. Use country code without + or 00.	Yes
name	Optional field of any string.	String	Yes
surname	Optional field of any string.	String	No
title	Optional field of any string.	String	No
address	Optional field of any string.	String	No
email	Optional field of any string.	String	No
active	Refers to contact status, available options "true" or "false".	String	Yes
Option1	Optional field of any string.	String	No
Option2	Optional field of any string.	String	No

customfields	Key value pair of custom field names and their respective values. Key-value should match the custom fields defined in Octapush.	CustomField {"FieldName":"FieldValue" , "FieldName2":"FieldValue 2" ….}	No

Request Example (POST)

```
{
    "id": "xxxxxxxx-xxxx-xxxx",
    "subscriberlistid": "xxxxxxxx-xxxx-xxxx",
    "phonenumber": "3069XXXXXXXX",
    "name": "ContactName",
    "surname": "ContactSurname",
    "title": "Mr",
    "address": "olympias 2",
    "email": "info@yuboto.com",
    "active": "true",
    "customfields": {"FieldName":"FieldValue"}
}
```

Request Example (GET)

1- Header Authentication:

```
https://services.yuboto.com/omni/v1/EditContact?id=xxxxxxxx-xxxx-
xxxx&subscriberlistid=xxxxxxxx-xxxx-
xxxx&phonenumber=3069XXXXXXXX&name=ContactName&surname=ContactSurname&title=M
r&address=olympias2&email=info@yuboto.com&active=true&customfields[FieldName]
=FieldValue
```

2- URL Authentication (apiKey Parameter Authentication Example):

```
https://services.yuboto.com/omni/v1/EditContact?id=xxxxxxxx-xxxx-
xxxx&subscriberlistid=xxxxxxxx-xxxx-
xxxx&phonenumber=3069XXXXXXXX&name=ContactName&surname=ContactSurname&title=M
r&address=olympias2&email=info@yuboto.com&active=true&customfields[FieldName]
=FieldValue&apiKey=XFv4RERCQTktNjBEQi00ODAyLUE3NTEtMTUyMDkzMsU4QkdFL
```

Response Example

```
{
    "ErrorCode": 0,
    "ErrorMessage": null,
    "Status": "updated",
    "key": "xxxxxxxx-xxxx-xxxx"
}
```

ErrorCode: The response error code for this call. This will be 0 if successful.

ErrorMessage: The response error message. This will be null if successful.

Status: The performed action status.

key: The unique ID of the updated record.

If the method called successfully then the ErrorCode has the value 0 and the ErrorMessage contains a zero-length string. If an error occurred then, consult the error message that appears.

12.5 Delete Contact

Description

DeleteContact method deletes an existing Contact.

URL to web service operation

```
https://services.yuboto.com/omni/v1/DeleteContact
```

Request Parameters

Variables	Description	Permitted Values	Required
id	Refers to the contact unique ID to be deleted.	String	Yes

Request Example (POST)

```
{
    "id": "xxxxxxxx-xxxx-xxxx",
}
```

Request Example (GET)

1- Header Authentication:

```
 https://services.yuboto.com/omni/v1/DeleteContact?id=xxxxxxxx-xxxx-xxxx
```

2- URL Authentication (apiKey Parameter Authentication Example):

```
https://services.yuboto.com/omni/v1/DeleteContact?id=xxxxxxxx-xxxx-
xxxx&apiKey=XFv4RERCQTktNjBEQi00ODAyLUE3NTEtMTUyMDkzMsU4QkdFL
```

Response Example

```
{
    "ErrorCode": 0,
    "ErrorMessage": null,
    "Status": "deleted",
    "key": "xxxxxxxx-xxxx-xxxx"
}
```

ErrorCode: The response error code for this call. This will be 0 if successful.

ErrorMessage: The response error message. This will be null if successful.

Status: The performed action status.

key: The unique ID of the deleted record.

If the method called successfully then the ErrorCode has the value 0 and the ErrorMessage contains a zero-length string. If an error occurred then, consult the error message that appears.

12.6 Get Custom Fields

Description

GetCustomFields method retrieves all existing custom fields created in Octapush platform.

URL to web service operation

```
https://services.yuboto.com/omni/v1/GetCustomFields
```

Request Example (POST)

No required parameters for this method.

GetCustomFields Example (GET)

1- Header Authentication:

```
 https://services.yuboto.com/omni/v1/GetCustomFields
```

2- URL Authentication (apiKey Parameter Authentication Example):

```
https://services.yuboto.com/omni/v1/GetCustomFields?apiKey=XFv4RERCQTktNjBEQi
00ODAyLUE3NTEtMTUyMDkzMsU4QkdFL
```

Response Example

```
{
    "ErrorCode": 0,
    "ErrorMessage": null,
    "names": [
        {
           "name":"CustomFieldName"
        }
]
```

ErrorCode: The response error code for this call. This will be 0 if successful.

ErrorMessage: The response error message. This will be null if successful.

Names: List of the names of the available custom fields.

If the method called successfully then the ErrorCode has the value 0 and the ErrorMessage contains a zero-length string. If an error occurred then, consult the error message that appears.

13. Blacklisted

13.1 Get Blacklisted

Description

GetBlacklisted method retrieves all blacklisted phone numbers.

URL to web service operation

```
https://services.yuboto.com/omni/v1/GetBlacklisted
```

Request Example (POST)

No required parameters for this method.

Request Example (GET)

1- Header Authentication:

```
 https://services.yuboto.com/omni/v1/GetBlacklisted
```

2- URL Authentication (apiKey Parameter Authentication Example):

```
https://services.yuboto.com/omni/v1/GetBlacklisted?apiKey=XFv4RERCQTktNjBEQi0
0ODAyLUE3NTEtMTUyMDkzMsU4QkdFL
```

Response Example

```
{
    "ErrorCode": 0,
    "ErrorMessage": null,
    "phonenumbers": [
           {
                 "69XXXXXXXX",
                 "69XXXXXXXX"
           }
      ]
}
```

ErrorCode: The response error code for this call. This will be 0 if successful.

ErrorMessage: The response error message. This will be null if successful.

phonenumbers: List of Blacklisted Phone Numbers.

If the method called successfully then the ErrorCode has the value 0 and the ErrorMessage contains a zero-length string. If an error occurred then, consult the error message that appears.

13.2 Create Blacklisted

Description

CreateBlacklisted method inserts a phone number to the blacklist.

URL to web service operation

```
https://services.yuboto.com/omni/v1/CreateBlackListed
```

Parameters

Variables	Description	Permitted Values	Required
phonenumber	Refers to the phone number of the recipient to be added as Blacklisted.	String/Array. Use country code without + or 00.	Yes

Request Example (POST)

```
{
    "phonenumber" : "3069XXXXXXXX"
}
```

Request Example (GET)

1- Header Authentication:

```
 https://services.yuboto.com/omni/v1/CreateBlackListed?phonenumber=69XXXXXXXX
```

2- URL Authentication (apiKey Parameter Authentication Example):

```
https://services.yuboto.com/omni/v1/CreateBlackListed?phonenumber=69XXXXXXXX&
apiKey=XFv4RERCQTktNjBEQi00ODAyLUE3NTEtMTUyMDkzMsU4QkdFL
```

Response Example

```
{
    "ErrorCode": 0,
    "ErrorMessage": null,
    "Status": "created",
    "key": "3069XXXXXXXX"
}
```

ErrorCode: The response error code for this call. This will be 0 if successful.

ErrorMessage: The response error message. This will be null if successful.

Status: The performed action status.

key: The phone number of the newly created record.

If the method called successfully then the ErrorCode has the value 0 and the ErrorMessage contains a zero-length string. If an error occurred then, consult the error message that appears.

13.3 Delete Blacklisted

Description

DeleteBlacklisted method removes a phone number from the blacklist.

URL to web service operation

```
https://services.yuboto.com/omni/v1/DeleteBlackListed
```

Parameters

Variables	Description	Permitted Values	Required
phonenumber	Refers to the phone number of the recipient to be removed from Blacklisted.	String. Do not use country code or + or 00.	Yes

Request Example Body (POST)

```
{
    "phonenumber" : "3069XXXXXXXX"
}
```

Request Example (GET)

1- Header Authentication:

```
https://services.yuboto.com/omni/v1/DeleteBlackListed?phonenumber=69XXXXXXXX
```

2- URL Authentication (apiKey Parameter Authentication Example):

```
https://services.yuboto.com/omni/v1/DeleteBlackListed?phonenumber=69XXXXXXXX&
apiKey=XFv4RERCQTktNjBEQi00ODAyLUE3NTEtMTUyMDkzMsU4QkdFL
```

Response example

```
{
    "ErrorCode": 0,
    "ErrorMessage": null,
    "Status": "deleted",
    "key": "3069XXXXXXXX"
}
```

ErrorCode: The response error code for this call. This will be 0 if successful.

ErrorMessage: The response error message. This will be null if successful.

Status: The performed action status.

key: The phone number of the removed record.

If the method called successfully then the ErrorCode has the value 0 and the ErrorMessage contains a zero-length string. If an error occurred then, consult the error message that appears.

14. Error Codes

Error Code	Error Description
0	Successful action
2	Invalid API key
3	Invalid channel
4	No credits

5	Routing settings not found for this channel. Please contact your administrator
6	This country is not supported for this channel. Please contact your administrator
7	Invalid Id
8	Invalid username
9	Invalid iso2 value
10	Country not supported. Please contact your administrator
11	Please select iso2 or phonenumber value
12	Invalid phone number
13	Invalid or not supported phone number
14	Invalid phone number
15	Invalid phone number
16	Time to send cannot be less than current time
17	callback_url value is too long. Max length 255 chars
18	option1 value is too long. Max length 50 chars
19	option2 value is too long. Max length 50 chars
20	Missing value phonenumbers.Select at least one valid recipient
22	Your Ip has been blocked due to several invalid attemps. Please contact your administrator
23	Access denied. Please contact your administrator
24	Your account is not active. Please contact your administrator
25	Your account has been blocked. Please contact your administrator
26	Your IP has been blocked due to several invalid attempts. Please contact your administrator
27	Submission already canceled
28	Submission already completed. It cannot be canceled
29	Submission is failed. It cannot be canceled
30	Submission in process. It cannot be canceled
31	Submission cannot be canceled
32	You are not allowed to perform this action. Please contact your administrator
33	Pin length out of range. Accepted values between 4 and 32
34	Invalid pin type. Accepted values numeric, alpha, alphanumeric
35	Expiration time out of range. Accepted values between 60 and 600 (seconds)
36	In order to use the TwoFa functionality text must include the tag {pin_code}.
37	Invalid pin
38	Pin expired
39	You do not have any active subscription in your account
40	callback_data value is too long. Max length 255 chars
41	Character ? is invalid in callback_url value
42	Character ? is invalid in callback_data value
43	Invalid credentials
44	Invalid credentials
45	Invalid credentials
100	SMS is not activated in your account. Please contact your administrator
102	Sms sender was not provided
103	Length of sms sender must be 16 digits numeric or 11 digits alphanumeric
104	Sms Text was not provided
105	Length of sms text is too long. Max Length 2000 chars

106	Sms validity out of range. Accepted values between 30 and 4320
107	Invalid smstype. Accepted values sms, flash, unicode
108	Sms Sender is not valid. Please contact your administrator
109	Sms Sender is rejected and it cannot be used
110	Sms Sender is in progress for approval. Please try again later
111	Sms Sender is blocked and it cannot be used
200	Viber is not activated in your account. Please contact your administrator
201	Invalid message type. Please select one of the following combinations: text, text-image-button-caption, text-button-action, image
202	Length of text is too long. Max Length 1000 chars
203	Length of expiryText is too long. Max Length 1000 chars
204	Length of buttonCaption is too long. Max Length 30 chars
205	Length of buttonAction is too long. Max Length 256 chars
206	Length of image is too long. Max Length 256 chars
207	Validity Out Of Range. Accepted values between 30 and 1209600
208	Invalid viber sender. Sender must be between 1 and 28 characters
209	Viber Sender is not valid. Please contact your administrator
210	ExpiryText is mandatory if there is validity
211	Invalid image file. Supported image files are JPG/JPEG/PNG
212	Viber Sender is canceled and it cannot be used
213	Viber Sender is rejected and it cannot be used
214	Viber Sender is in progress for approval. Please try again later
215	Viber Sender is pending for approval. Please try again later
216	Viber Sender is waiting for approval. Please try again later
217	Your account settings is allowed to send only transactional messages. Please send only text.
218	Length of fileName is too long. Max Length 25 chars
219	FileName is not supported. Please check your documentation for the supported file extentions
220	Viber Sender does not support this type of message
221	Missing fileType Value
222	Missing fileName Value
223	ServiceType is not defnied. Multiple senderIds require ServiceType definition.
224	Invalid ServiceType defined. Value should be a number between 1 to 3 (1: OneWay, 2: TwoWay, 3: Session)
225	No SenderId found for the selected ServiceType
226	Multiple SenderIds found for the selected ServiceType
228	Video is missing the thumbnail
229	Length of thumbnail is too long. Max length 1000 characters
230	The size of the Video file exceeds the limit of 200MB
231	The duration of the Video file exceeds the limit of 600 seconds
232	OTP Service only available for single phone number
300	Omni is not activated in your account. Please contact your administrator
301	Invalid omni_channel
302	Duplicate channel SMS
303	Duplicate channel VIBER

304	Select at least one channel
305	Invalid Sms channel priority
306	Invalid Viber channel priority
307	Invalid channel priority
308	No dlr for the selected phonenumber
400	Invalid datein value
401	Invalid datein. No available data before the year 2013
402	Invalid datein. The parameter cannot be later than the present date
403	Invalid phonenumber value
500	A system error has occured. Please contact your administrator
501	Invalid Service Endpoint. Please check service documentation or contact your administrator
503	GET method is not allowed. Please use POST method instead
600	No records found for Contact Subscriber list
601	@"Field ""name"" cannot be empty"
602	@"Field ""name"" must be less or equal to 50 characters"
603	Subscriber list not found
604	@"Field ""id"" cannot be empty"
605	@"Field ""subscriberlistid"" cannot be empty"
606	Contact not found
607	No records found for Blacklisted Phone Numbers
608	Phone Number already Blacklisted
609	Phone Number not is not Blacklisted
610	@"Field ""phonenumber"" cannot be empty"},
611	@"Field ""phonenumber"" must be less or equal to 50 characters"
612	@"Field ""surname"" cannot be empty"
613	@"Field ""surname"" must be less or equal to 50 characters"
614	Phone Number should not include any characters
615	No records found for this Subscriber list
616	@"Field ""active"" cannot be empty"},
617	@"Field ""active"" only accepts values ""true"" or ""false"""},
618	@"Invalid custom fields format: Format allowed example : { ""key"": ""value1"" , ""key2"": ""value2"" , ""key3"": ""value3"" ....}"},
619	@"Optional fields must be less or equal to 1000 characters"
620	@"Minimum allowed number of Subscriber lists for search is 1."
621	@"PageSize must a any value between 1 and 1000."
622	@"Unknown Field {0}."
623	Contact Subscriber list '{0}' not found
624	No records found for this search
625	No records found for custom fields