Table of content

Getting started

Introduction

This page will help you get started with SMS API. You'll be up and running in a jiffy!

Welcome to SMS API documentation!

This document will provide instructions on how to quickly integrate SMS messaging services into various solutions by using HTTP application programming interface (HTTP API). The HTTP API can be used for sending SMS messages, collecting delivery reports, making Number Context (number validation) requests and receiving inbound SMS messages sent from mobile phones.

API is based on REST standards, enabling you to use your browser for accessing URLs. In order to interact with our API, any HTTP client in any programming language can be used.

Base URL

Submit all requests to the base URL. All the requests are submitted thorough HTTP `POST` or `GET` method.

Base URL: `http://107.20.199.106`

Content-Type & Accept header

SMS API supports `JSON` and `XML` Content-Types and Accept criteria that should be specified in the header. If the Content-Type is not specified you will receive a General error. Depending which Accept type is chosen in the deader for the request, the same one will be applied in the response.

Content-Type: `application/json` or `application/xml`.

Accept header: `application/json` or `application/xml`.

Authorization

We support basic authorization using a username and password with Base64 encoding variation RFC2045-MIME.

The authorization header is constructed as follows:

1. Username and password are combined into a string `username:password`.
2. The resulting string is encoded using the RFC2045-MIME variant of Base64.
3. The authorization method and a space, like this: `"Basic "`, are put before the encoded string.

> Example:
>
> Username: `Aladdin`
> Password: `open sesame`
>
> Base64 encoded string: `QWxhZGRpbjpvcGVuIHNlc2FtZQ==`
>
> Authorization header: `Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==`

Send SMS

Send your first SMS using API!

In a few simple steps, we will explain how to send an SMS using HTTP API.

Your username and password has to be encoded in `base64` like this:

- Combine the username and password into a string `username:password`.
- Encode the resulting string using Base64 encoder.

> **Example:**
>
> Username: `Aladdin`
> Password: `open sesame`
>
> String: `Alladin:open sesame`
> Base64 encoded string: `QWxhZGRpbjpvcGVuIHNlc2FtZQ==`

The message will be sent only to a valid phone number (numbers), written in **international format** e.g. `41793026727` .

> **Phone numbers format**
>
> We strongly recommend using the E.164 number formatting. E.164 numbers are internationally standardized to a fifteen digit maximum length. Phone numbers are usually prefixed with + (plus sign), followed by a *country code*, *network code* and the *subscriber number*. Phone numbers that are not E.164 formatted may work, depending on the handset or network.

Now, you are ready to create a HTTP POST request to `http://107.20.199.106/restapi/sms/1/text/single`

Your **Header** should contain *authorization* and *content type*:

- Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
- Content-Type: application/json

Request body contains the message you wish to send with `from`, `to` and `text` parameters.

Full **JSON request** is shown below:

```
POST /restapi/sms/1/text/single HTTP/1.1
Host: 107.20.199.106
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
Content-Type: application/json
Accept: application/json
```

```
{
   "from":"InfoSMS",
   "to":"41793026727",
   "text":"My first SMS"
}
```

That's it! You should receive an SMS in a few moments.

Send SMS response

Handle SMS API response.

After the "Send SMS" HTTP request was submitted to the SMS API, you will get a response containing some useful information. If everything went well, it should provide an **200 OK** response with message details in the response body.

Here is an example of a request for sending a single SMS:

```
POST /restapi/sms/1/text/single HTTP/1.1
Host: 107.20.199.106
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
Content-Type: application/json
Accept: application/json

{
   "from":"InfoSMS",
   "to":"41793026727",
   "text":"My first SMS"
}
```

And the appropriate response is shown below:

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

{
   "messages":[
      {
         "to":"41793026727",
```

```
            "status":{
               "id":0,
               "groupId":0,
               "groupName":"ACCEPTED",
               "name":"MESSAGE_ACCEPTED",
               "description":"Message accepted"
            },
            "smsCount":1,
            "messageId":"2250be2d4219-3af1-78856-aabe-1362af1edfd2"
         }
      ]
   }
```


- **messages** is an array of all SMS messages that were sent in the last request. In our case, it contains only one message
- **to** is a phone number which you have sent the SMS message to
- Each message successfully submitted to the platform is uniquely identified with the **messageId**. Furthermore, the Message ID can be used for checking Delivery status or Sent messages logs
- **smsCount** is the number of parts the message was split into
- **status** is the object that further describes the state of sent message.

Getting delivery reports

Check if your messages were successfully delivered.

After you have sent a couple of messages, you are able to check if they were successfully delivered by making this request:

```
GET  http://107.20.199.106/restapi/sms/1/reports
```

Available **query parameters** are:

- **bulkId:** The ID uniquely identifies the sent SMS request. This filter will enable you to receive delivery reports for all the messages using just one request.
- **messageId:** The ID that uniquely identifies the message sent.
- **limit:** The maximum number of delivery reports you want to get.

As a response, you will get a collection of unread delivery reports.

> **Important:**

> Delivery reports can only be retrieved one time. Once you retrieve a delivery report, you will not be able to get the same report again by using this endpoint.

Here is the JSON request example for **getting reports without any query parameter:**

```
GET /restapi/sms/1/reports HTTP/1.1
Host: 107.20.199.106
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
Accept: application/json
```

Below you can see the response to delivery reports request:

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

{
   "results":[
      {
         "bulkId":"80664c0c-e1ca-414d-806a-5caf146463df",
         "messageId":"bcfb828b-7df9-4e7b-8715-f34f5c61271a",
         "to":"38598111",
         "sentAt":"2015-02-12T09:51:43.123+0100",
         "doneAt":"2015-02-12T09:51:43.127+0100",
         "smsCount":1,
         "price":{
            "pricePerMessage":0.01,
            "currency":"EUR"
         },
         "status":{
            "groupId":3,
            "groupName":"DELIVERED",
            "id":5,
            "name":"DELIVERED_TO_HANDSET",
            "description":"Message delivered to handset"
         },
         "error":{
            "groupId":0,
            "groupName":"OK",
            "id":0,
            "name":"NO_ERROR",
            "description":"No Error",
            "permanent":false
         }
```

```
        },
        {
            "bulkId":"08fe4407-c48f-4d4b-a2f4-9ff583c985b8",
            "messageId":"12db39c3-7822-4e72-a3ec-c87442c0ffc5",
            "to":"385981112",
            "sentAt":"2015-02-12T09:50:22.221+0100",
            "doneAt":"2015-02-12T09:50:22.232+0100",
            "smsCount":1,
            "price":{
                "pricePerMessage":0.01,
                "currency":"EUR"
            },
            "status":{
                "groupId":3,
                "groupName":"DELIVERED",
                "id":5,
                "name":"DELIVERED_TO_HANDSET",
                "description":"Message delivered to handset"
            },
            "error":{
                "groupId":0,
                "groupName":"OK",
                "id":0,
                "name":"NO_ERROR",
                "description":"No Error",
                "permanent":false
            }
        }
    ]
}
```


In a response, you will receive an array of `results` which contain:

- `to` represents the recipient's phone number. This way you can connect a delivery report to a phone number.
- `bulkId` and `messageId`], the ids that uniquely identify the request and the messages sent.
- `sentAt` and `doneAt`
- `smsCount` represents number of messages
- `price` object with `pricePerMessage` and `currency` parameters
- `status` and `error` objects

> **Note:**
>
> If you try making this same request again, you will get an empty set because all delivery reports were

> read:

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

{
   "results":[]
}
```

If you send a mass number of messages but you are only interested in seeing the delivery report for only one, just set a query parameter in the request.

Append `?messageId=ff4804ef-6ab6-4abd-984d-ab3b1387e852` on the request url, and you will get delivery report only for that message.

Besides the **messageId**, you can use **bulkId** or simply set the **limit** on the number of reports you wish to retrieve. Here is the JSON request example for getting the reports with query parameter:

```
GET /restapi/sms/1/reports?messageId=ff4804ef-6ab6-4abd-984d-ab3b1387e852
HTTP/1.1
Host: 107.20.199.106
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
Accept: application/json
```

The following JSON will be given as a response:

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

{
   "results":[
      {
         "bulkId":"8c20f086-d82b-48cc-b2b3-3ca5f7aca9fb",
         "messageId":"ff4804ef-6ab6-4abd-984d-ab3b1387e852",
         "to":"385981178",
         "sentAt":"2015-02-12T09:58:20.323+0100",
         "doneAt":"2015-02-12T09:58:20.337+0100",
         "smsCount":1,
         "price":{
            "pricePerMessage":0.01,
            "currency":"EUR"
         },
```

```
            "status":{
               "id":5,
               "groupId":3,
               "groupName":"DELIVERED",
               "name":"DELIVERED_TO_HANDSET",
               "description":"Message delivered to handset"
            },
            "error":{
               "groupId":0,
               "groupName":"OK",
               "id":0,
               "name":"NO_ERROR",
               "description":"No Error",
               "permanent":false
            }
         }
      ]
   }
```


As you can see, that message was successfully delivered without any error.

The opposite to one time delivery reports are **logs** which can be used to see the history for all the messages that you have sent. In the next step of this tutorial, we are going to show you how to get logs using our API.

Additionally, you are able to setup an end-point on your callback server so you can receive a **Delivery reports on Notify URL**.

Getting SMS logs

Your sent SMS message history.

Logs with sent SMS message history can be requested for all messages by using a single request: `GET http://107.20.199.106/restapi/sms/1/logs` .
Unlike delivery reports, these logs can be requested as many times as you want.

Let's see what happens when you request all of your logs, without any query parameter:

```
GET /restapi/sms/1/logs HTTP/1.1
Host: 107.20.199.106
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
```

```
Accept: application/json
```

As a response, you will get the following result:

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

{
   "results":[
      {
         "bulkId":"bafdeb3d-719b-4cce-8762-54d47b40f3c5",
         "messageId":"07e03aae-fabc-44ad-b1ce-222e14094d70",
         "to":"41793026727",
         "from":"InfoSMS",
         "text":"Test SMS.",
         "sentAt":"2015-02-23T17:41:11.833+0100",
         "doneAt":"2015-02-23T17:41:11.843+0100",
         "smsCount":1,
         "mccmnc":"22801",
         "price":{
            "pricePerMessage":0.01,
            "currency":"EUR"
         },
         "status":{
            "groupId":3,
            "groupName":"DELIVERED",
            "id":5,
            "name":"DELIVERED_TO_HANDSET",
            "description":"Message delivered to handset"
         },
         "error":{
            "groupId":0,
            "groupName":"OK",
            "id":0,
            "name":"NO_ERROR",
            "description":"No Error",
            "permanent":false
         }
      },
      {
         "bulkId":"06479ba3-5977-47f6-9346-fee0369bc76b",
         "messageId":"1f21d8d7-f306-4f53-9f6e-eddfce9849ea",
         "to":"41793026727",
         "from":"InfoSMS",
```

```
            "text":"Test SMS.",
            "sentAt":"2015-02-23T17:40:31.773+0100",
            "doneAt":"2015-02-23T17:40:31.787+0100",
            "smsCount":1,
            "mccmnc":"22801",
            "price":{
               "pricePerMessage":0.01,
               "currency":"EUR"
            },
            "status":{
               "groupId":3,
               "groupName":"DELIVERED",
               "id":5,
               "name":"DELIVERED_TO_HANDSET",
               "description":"Message delivered to handset"
            },
            "error":{
               "groupId":0,
               "groupName":"OK",
               "id":0,
               "name":"NO_ERROR",
               "description":"No Error",
               "permanent":false
            }
         }
      ]
   }
```


Logs carry similar information as delivery reports, with some added fields.

> **Important:**
>
> **SMS logs are available for the last 48 hours!**

Since this logs example was for all the messages you have sent over the platform for the last **48 hours**, you might need some filters to search through them. The filters you can use are:

Parameter	Type	Description
from	String	Sender address.
to	String	Destination address.
bulkId	String[]	Bulk ID for which logs are requested.

messageId	String[]	Message ID for which logs are requested.
generalStatus	String	Sent SMS status.
sentSince	Date	Lower limit on date and time of sending SMS.
sentUntil	Date	Upper limit on date and time of sending SMS.
limit	int	Max number of messages in returned logs.
mcc	String	Mobile country code.
mnc	String	Mobile network code.

Now, let's try getting **logs with "from", "to" and "limit" as filters**:

```
GET /restapi/sms/1/logs?from=InfoSMS&to=41793026727&limit=1 HTTP/1.1
Host: 107.20.199.106
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
Accept: application/json
```

The response will be:

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

{
   "results":[
      {
         "bulkId":"82d1d36e-e4fb-4194-8b93-caeb053bd327",
         "messageId":"fc0cbfb8-7a72-40da-a76d-e2c2d9400835",
         "to":"41793026727",
         "from":"InfoSMS",
         "text":"Test SMS.",
         "sentAt":"2015-02-23T17:42:05.390+0100",
         "doneAt":"2015-02-23T17:42:05.390+0100",
         "smsCount":1,
         "mccmnc":"22801",
         "price":{
            "pricePerMessage":0,
            "currency":"EUR"
         },
         "status":{
            "groupId":5,
```

```
            "groupName":"REJECTED",
            "id":6,
            "name":"REJECTED_NETWORK",
            "description":"Network is forbidden",
            "action":"Contact account manager"
         },
         "error":{
            "groupId":0,
            "groupName":"OK",
            "id":0,
            "name":"NO_ERROR",
            "description":"No Error",
            "permanent":false
         }
      }
   ]
}
```


Number Context

The API which ensures mobile number validity!

Number Context helps you keep your mobile numbers database up to date.

Mobile subscribers often change numbers, go into roaming and change providers while retaining their original phone number. Knowing which mobile numbers are in use and available, or which network your client is currently using can greatly improve accuracy and cost effectiveness for many types of businesses.

With Number Context, you can determine:

- which numbers are currently active
- is the mobile number in roaming
- is the mobile number ported
- the optimal route for messages and voice
- the type of number (e.g. land-line, machine-to-machine, mobile etc.)

The following example shows how you can get Number Context information using our API:

```
POST /restapi/number/1/query HTTP/1.1
Host: 107.20.199.106
```

```
Authorization: Basic dXNlcm5hbWU6cGFzc3dvcmQ=
Content-Type: application/json
Accept: application/json

{
   "to":["41793026727"]
}
```

The `to` parameter is a list of all the numbers you want to check.

Here is your result:

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

{
   "ncResults":[
      {
         "to":"41793026727",
         "mccMnc":"22801",
         "imsi":"228012120181810",
         "originalNetwork":{
            "networkPrefix":"79",
            "countryPrefix":"41"
         },
         "ported":false,
         "roaming":false,
         "status":{
            "groupId":3,
            "groupName":"DELIVERED",
            "id":5,
            "name":"DELIVERED_TO_HANDSET",
            "description":"Message delivered to handset"
         },
         "error":{
            "groupId":0,
            "groupName":"OK",
            "id":0,
            "name":"NO_ERROR",
            "description":"No Error",
            "permanent":false
         }
      }
   ]
```

```
}
```

Response codes

SMS API statuses and GSM codes.

Check the list of response codes for statuses and GSM errors which could be received.

Status object example:

```
{
   "groupId":3,
   "groupName":"DELIVERED",
   "id":5,
   "name":"DELIVERED_TO_HANDSET",
   "description":"Message delivered to handset"
}
```

Statuses groups

GroupId	GroupName	Description
0	ACCEPTED	Message is accepted.
1	PENDING	Message is in pending status.
2	UNDELIVERABLE	Message is undeliverable.
3	DELIVERED	Message is delivered.
4	EXPIRED	Message is expired.
5	REJECTED	Message is rejected.

Statuses

Id	GroupId	Name	Description	Action
1	1	PENDING_TIME_VIOLATION	Time window violation	NULL
2	3	DELIVERED_TO_OPERATOR	Message delivered to operator	NULL
3	1	PENDING_WAITING_DELIVERY	Message sent, waiting for delivery report	NULL
4	2	UNDELIVERABLE_REJECTED_OPERATOR	Message rejected by operator	NULL
5	3	DELIVERED_TO_HANDSET	Message delivered to handset	NULL
6	5	REJECTED_NETWORK	Network is forbidden	Contact account manager
7	1	PENDING_ENROUTE	Message sent to next instance	NULL
8	5	REJECTED_PREFIX_MISSING	Number prefix missing	NULL
9	2	UNDELIVERABLE_NOT_DELIVERED	Message sent not delivered	NULL
10	5	REJECTED_DND	Destination on DND list	NULL
11	5	REJECTED_SOURCE	Invalid Source address	NULL
12	5	REJECTED_NOT_ENOUGH_CREDITS	Not enough credits	NULL
13	5	REJECTED_SENDER	By Sender	Remove sender from blacklist
14	5	REJECTED_DESTINATION	By Destination	Remove destination from blacklist

15	4	EXPIRED_EXPIRED	Message expired	NULL
16	5	REJECTED_NOT_REACHABLE	Network not reachable	NULL
17	5	REJECTED_PREPAID_PACKAGE_EXPIRED	Prepaid package expired	Top-Up your account to extend the validty period
18	5	REJECTED_DESTINATION_NOT_REGISTERED	Destination not registered	NULL
19	5	REJECTED_ROUTE_NOT_AVAILABLE	Route not available	Contact account manager
20	5	REJECTED_FLOODING_FILTER	Rejected flooding	STOP SPAMMING
21	5	REJECTED_SYSTEM_ERROR	System error	NULL
22	4	EXPIRED_UNKNOWN	Unknown Reason	NULL
23	5	REJECTED_DUPLICATE_MESSAGE_ID	Rejected duplicate message ID	NULL
24	5	REJECTED_INVALID_UDH	Rejected invalid UDH	NULL
25	5	REJECTED_MESSAGE_TOO_LONG	Rejected message too long	NULL

Error object example:


```
{
   "groupId":0,
   "groupName":"OK",
   "id":0,
   "name":"NO_ERROR",
   "description":"No Error",
   "permanent":false
}
```

Errors Groups

GroupId	GroupName	Description
0	OK	No error.
1	HANDSET_ERRORS	Handset error occurred.
2	USER_ERRORS	User error occurred.
3	OPERATOR_ERRORS	Operator error occurred.

GSM Error Codes

Id	Short description	Is permanent
0	NO_ERROR	NULL
1	EC_UNKNOWN_SUBSCRIBER	1
5	EC_UNIDENTIFIED_SUBSCRIBER	0
6	EC_ABSENT_SUBSCRIBER_SM	0
9	EC_ILLEGAL_SUBSCRIBER	1
10	EC_BEARER_SERVICE_NOT_PROVISIONED	0
11	EC_TELESERVICE_NOT_PROVISIONED	1
12	EC_ILLEGAL_EQUIPMENT	1
13	EC_CALL_BARRED	0
20	EC_SS_INCOMPATIBILITY	0
21	EC_FACILITY_NOT_SUPPORTED	0
27	EC_ABSENT_SUBSCRIBER	0
31	EC_SUBSCRIBER_BUSY_FOR_MT_SMS	0
32	EC_SM_DELIVERY_FAILURE	0

33	EC_MESSAGE_WAITING_LIST_FULL	0
34	EC_SYSTEM_FAILURE	0
35	EC_DATA_MISSING	1
36	EC_UNEXPECTED_DATA_VALUE	1
51	EC_RESOURCE_LIMITATION	0
71	EC_UNKNOWN_ALPHABET	1
72	EC_USSD_BUSY	1
255	EC_UNKNOWN_ERROR	1
256	EC_SM_DF_memoryCapacityExceeded	0
257	EC_SM_DF_equipmentProtocolError	0
258	EC_SM_DF_equipmentNotSM_Equipped	0
259	EC_SM_DF_unknownServiceCentre	0
260	EC_SM_DF_sc_Congestion	0
261	EC_SM_DF_invalidSME_Address	0
262	EC_SM_DF_subscriberNotSC_Subscriber	0
500	EC_PROVIDER_GENERAL_ERROR	0
502	EC_NO_RESPONSE	0
503	EC_SERVICE_COMPLETION_FAILURE	0
504	EC_UNEXPECTED_RESPONSE_FROM_PEER	0
507	EC_MISTYPED_PARAMETER	0
508	EC_NOT_SUPPORTED_SERVICE	0
509	EC_DUPLICATED_INVOKE_ID	0
511	EC_INITIATING_RELEASE	0
1024	EC_OR_appContextNotSupported	0
1025	EC_OR_invalidDestinationReference	0

1026	EC_OR_invalidOriginatingReference	0
1027	EC_OR_encapsulatedAC_NotSupported	0
1028	EC_OR_transportProtectionNotAdequate	0
1029	EC_OR_noReasonGiven	0
1030	EC_OR_potentialVersionIncompatibility	0
1031	EC_OR_remoteNodeNotReachable	0
1152	EC_NNR_noTranslationForAnAddressOfSuchNature	0
1153	EC_NNR_noTranslationForThisSpecificAddress	0
1154	EC_NNR_subsystemCongestion	0
1155	EC_NNR_subsystemFailure	0
1156	EC_NNR_unequippedUser	0
1157	EC_NNR_MTPfailure	0
1158	EC_NNR_networkCongestion	0
1159	EC_NNR_unqualified	0
1160	EC_NNR_errorInMessageTransportXUDT	0
1161	EC_NNR_errorInLocalProcessingXUDT	0
1162	EC_NNR_destinationCannotPerformReassemblyXUDT	0
1163	EC_NNR_SCCPfailure	0
1164	EC_NNR_hopCounterViolation	0
1165	EC_NNR_segmentationNotSupported	0
1166	EC_NNR_segmentationFailure	0
1281	EC_UA_userSpecificReason	0
1282	EC_UA_userResourceLimitation	0
1283	EC_UA_resourceUnavailable	0
1284	EC_UA_applicationProcedureCancellation	0

1536	EC_PA_providerMalfunction	0
1537	EC_PA_supportingDialogOrTransactionRealeased	0
1538	EC_PA_ressourceLimitation	0
1539	EC_PA_maintenanceActivity	0
1540	EC_PA_versionIncompatibility	0
1541	EC_PA_abnormalMapDialog	0
1792	EC_NC_abnormalEventDetectedByPeer	0
1793	EC_NC_responseRejectedByPeer	0
1794	EC_NC_abnormalEventReceivedFromPeer	0
1795	EC_NC_messageCannotBeDeliveredToPeer	0
1796	EC_NC_providerOutOfInvoke	0
2048	EC_TIME_OUT	0
2049	EC_IMSI_BLACKLISTED	1
2050	EC_DEST_ADDRESS_BLACKLISTED	1
2051	EC_InvalidMscAddress	0
4096	EC_invalidPduFurmat	1
4097	EC_NotSubmittedToGMSC	1
4100	EC_Cancelled	1
4101	EC_ValidityExpired	1
4102	EC_NotSubmittedToSmppChannel	0

Send SMS

Single textual message

This method allows you to send a single textual message to one or more destination addresses.

Definition

`http://107.20.199.106/restapi/sms/1/text/single`

Parameters

Parameter	Type	Description
from	String	Represents sender ID and it can be alphanumeric or numeric. *Alphanumeric* sender ID length should be between 3 and 11 characters (Example: `CompanyName`). *Numeric* sender ID length should be between 3 and 14 characters.
to	String[]	**Required.** Array of message destination addresses. If you want to send a message to one destination, a single String is supported instead of an Array. Destination addresses must be in international format (Example: `41793026727`).
text	String	Text of the message that will be sent.

Examples

```
POST /restapi/sms/1/text/single HTTP/1.1
Host: 107.20.199.106
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
Content-Type: application/json
Accept: application/json

{
   "from":"InfoSMS",
   "to":"41793026727",
   "text":"Test SMS."
}
```

Result Format

```
{
   "messages":[
      {
         "to":"41793026727",
         "status":{
            "groupId":0,
            "groupName":"ACCEPTED",
            "id":0,
            "name":"MESSAGE_ACCEPTED",
            "description":"Message accepted"
         },
         "smsCount":1,
         "messageId":"2250be2d4219-3af1-78856-aabe-1362af1edfd2"
      }
   ]
}
```


Responseformat

On success, response header HTTP status code will be `200 OK` and the message will be sent.

If you try to send message without authorization, you will receive an error `401 Unauthorized`.

SMSResponse

Parameter	Type	Description
bulkId	String	The ID that uniquely identifies the request. Bulk ID will be received only when you send a message to more than one destination address.
messages	SMSResponseDetails[]	Array of sent message objects, one object per every message.

SMSResponseDetails

Parameter	Type	Description
to	String	The message destination address.
status	Status	Indicates whether the message is successfully sent, not sent, delivered, not delivered, waiting for delivery or any other possible status.

smsCount	int	The number of sent message segments.
messageId	String	The ID that uniquely identifies the message sent.

Status

Parameter	Type	Description
groupId	int	Status group ID.
groupName	String	Status group name.
id	int	Status ID.
name	String	Status name.
description	String	Human readable description of the status.
action	String	Action that should be taken to eliminate the error.

> **Long SMS:**
>
> Maximum length for one message is 160 characters for GSM7 standard or 70 characters Unicode encoded messages. If you send text which exceeds the maximum number of supported characters for one message, the sent message will be segmented and charged accordingly. One Long SMS that consists of two SMS counts as two SMS.

Additional examples

Single textual message to one destination

Request

JSON

```
POST /restapi/sms/1/text/single HTTP/1.1
Host: 107.20.199.106
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
Content-Type: application/json
Accept: application/json
```

```
{
   "from":"InfoSMS",
   "to":"41793026727",
   "text":"Test SMS."
}
```

XML

```
POST /restapi/sms/1/text/single HTTP/1.1
Host: 107.20.199.106
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
Content-Type: application/xml
Accept: application/xml

<request>
   <from>InfoSMS</from>
   <to>
      <to>41793026727</to>
   </to>
   <text>Test SMS.</text>
</request>
```

cURL

```
curl -X POST \
 -H 'Content-Type: application/json' \
 -H 'Accept: application/json' \
 -H 'Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==' \
 -d '{
   "from":"InfoSMS",
   "to":"41793026727",
   "text":"Test SMS."
}'  http://107.20.199.106/restapi/sms/1/text/single
```

PHP

```
<?php

$request = new HttpRequest();
$request->setUrl('http://107.20.199.106/restapi/sms/1/text/single');
$request->setMethod(HTTP_METH_POST);
```

```
$request->setHeaders(array(
  'accept' => 'application/json',
  'content-type' => 'application/json',
  'authorization' => 'Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ=='
));

$request->setBody('{
   "from":"InfoSMS",
   "to":"41793026727",
   "text":"Test SMS."
}');

try {
  $response = $request->send();

  echo $response->getBody();
} catch (HttpException $ex) {
  echo $ex;
}
```

Ruby

```
require 'uri'
require 'net/http'

url = URI("http://107.20.199.106/restapi/sms/1/text/single")

http = Net::HTTP.new(url.host, url.port)
http.use_ssl = true
http.verify_mode = OpenSSL::SSL::VERIFY_NONE

request = Net::HTTP::Post.new(url)
request["authorization"] = 'Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ=='
request["content-type"] = 'application/json'
request["accept"] = 'application/json'

request.body = "{\"from\":\"InfoSMS\",\"to\":\"41793026727\",
\"text\":\"Test SMS.\"}"

response = http.request(request)
puts response.read_body
```

Python

```
conn = http.client.httpConnection("107.20.199.106")

payload = "{\"from\":\"InfoSMS\",\"to\":\"41793026727\",\"text\":\"Test
SMS.\"}"

headers = {
    'authorization': "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==",
    'content-type':  "application/json",
    'accept': "application/json"
    }

conn.request("POST", "/restapi/sms/1/text/single", payload, headers)

res = conn.getresponse()
data = res.read()

print(data.decode("utf-8"))
```

Java

```
HttpResponse<String> response =
Unirest.post("http://107.20.199.106/restapi/sms/1/text/single")
  .header("authorization", "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==")
  .header("content-type",  "application/json")
  .header("accept",  "application/json")
  .body("{\"from\":\"InfoSMS\",\"to\":\"41793026727\",\"text\":\"Test
SMS.\"}")
  .asString();
```

C#

```
var client = new
RestClient("http://107.20.199.106/restapi/sms/1/text/single");

var request = new RestRequest(Method.POST);
request.AddHeader("accept", "application/json");
request.AddHeader("content-type",   "application/json");
request.AddHeader("authorization",  "Basic  QWxhZGRpbjpvcGVuIHNlc2FtZQ==");
request.AddParameter("application/json",   "{\"from\":\"InfoSMS\",
\"to\":\"41793026727\",\"text\":\"Test SMS.\"}",
ParameterType.RequestBody);
```

```
IRestResponse response = client.Execute(request);
```

JavaScript

```
var data = JSON.stringify({
  "from": "InfoSMS",
  "to": "41793026727",
  "text": "Test SMS."
});

var xhr = new XMLHttpRequest();
xhr.withCredentials = true;

xhr.addEventListener("readystatechange", function () {
  if (this.readyState === this.DONE) {
    console.log(this.responseText);
  }
});

xhr.open("POST", "http://107.20.199.106/restapi/sms/1/text/single");
xhr.setRequestHeader("authorization", "Basic
QWxhZGRpbjpvcGVuIHNlc2FtZQ==");
xhr.setRequestHeader("content-type", "application/json");
xhr.setRequestHeader("accept",  "application/json");

xhr.send(data);
```

Response

JSON

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

{
   "messages":[
      {
         "to":"41793026727",
         "status":{
            "groupId":0,
            "groupName":"Queue/Accepted",
            "id":0,
            "name":"MESSAGE_ACCEPTED",
```

```
            "description":"Message accepted"
         },
         "smsCount":1,
         "messageId":"2250be2d4219-3af1-78856-aabe-1362af1edfd2"
      }
   ]
}
```

XML

```
HTTP/1.1 200 OK
Content-Type: application/xml

<SMSResponse>
   <messages>
      <messages>
         <to>41793026727</to>
         <status>
            <id>0</id>
            <groupId>0</groupId>
            <groupName>ACCEPTED</groupName>
            <name>MESSAGE_ACCEPTED</name>
            <description>Message  accepted</description>
         </status>
         <smsCount>1</smsCount>
         <messageId>2250be2d4219-3af1-78856-aabe-1362af1edfd2</messageId>
      </messages>
   </messages>
</SMSResponse>
```

Single textual message to multiple destinations

Request

JSON

```
POST /restapi/sms/1/text/single HTTP/1.1
Host: 107.20.199.106
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
Content-Type: application/json
Accept: application/json
```

```
{
   "from":"InfoSMS",
   "to":[
      "41793026727",
      "41793026834"
   ],
   "text":"Test SMS."
}
```

XML

```
POST /restapi/sms/1/text/single HTTP/1.1
Host: 107.20.199.106
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
Content-Type: application/xml
Accept: application/xml

<request>
   <from>InfoSMS</from>
   <to>
      <to>41793026727</to>
      <to>41793026834</to>
   </to>
   <text>Test SMS.</text>
</request>
```

cURL

```
curl -X POST
 -H 'Content-Type: application/json'
 -H 'Accept: application/json'
 -H 'Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ=='
 -d '{
   "from":"InfoSMS",
   "to":[
      "41793026727",
      "41793026834"
   ],
   "text":"Test SMS."
}'  http://107.20.199.106/restapi/sms/1/text/single
```

PHP

```php
<?php

$request = new HttpRequest();
$request->setUrl('http://107.20.199.106/restapi/sms/1/text/single');
$request->setMethod(HTTP_METH_POST);

$request->setHeaders(array(
  'accept' => 'application/json',
  'content-type' => 'application/json',
  'authorization' => 'Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ=='
));

$request->setBody('{
   "from":"InfoSMS",
   "to":[
      "41793026727",
      "41793026834"
   ],
   "text":"Test SMS."
}');

try {
  $response = $request->send();

  echo $response->getBody();
} catch (HttpException $ex) {
  echo $ex;
}
```

Ruby

```ruby
require 'uri'
require 'net/http'

url = URI("http://107.20.199.106/restapi/sms/1/text/single")

http = Net::HTTP.new(url.host, url.port)
http.use_ssl = true
http.verify_mode = OpenSSL::SSL::VERIFY_NONE

request = Net::HTTP::Post.new(url)
request["authorization"] = 'Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ=='
request["content-type"] = 'application/json'
```

```
request["accept"] = 'application/json'

request.body = "{\"from\":\"InfoSMS\",\"to\":[\"41793026727\",
\"41793026834\"],\"text\":\"Test  SMS.\"}"

response =http.request(request)
puts response.read_body
```

Python

```
import http.client

conn = http.client.httpConnection("107.20.199.106")

payload = "{\"from\":\"InfoSMS\",\"to\":[\"41793026727\",
\"41793026834\"],\"text\":\"Test  SMS.\"}"

headers = {
    'authorization': "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==",
    'content-type':  "application/json",
    'accept': "application/json"
    }

conn.request("POST", "/restapi/sms/1/text/single", payload, headers)

res = conn.getresponse()
data = res.read()

print(data.decode("utf-8"))
```

Java

```
HttpResponse<String> response =
Unirest.post("http://107.20.199.106/restapi/sms/1/text/single")
  .header("authorization", "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==")
  .header("content-type",  "application/json")
  .header("accept",  "application/json")
  .body("{\"from\":\"InfoSMS\",\"to\":[\"41793026727\",
\"41793026834\"],\"text\":\"Test  SMS.\"}")
  .asString();
```

C#

```
var client = new
RestClient("http://107.20.199.106/restapi/sms/1/text/single");

var request = new RestRequest(Method.POST);
request.AddHeader("accept", "application/json");
request.AddHeader("content-type",  "application/json");
request.AddHeader("authorization",  "Basic  QWxhZGRpbjpvcGVuIHNlc2FtZQ==");
request.AddParameter("application/json",  "{\"from\":\"InfoSMS\",  \"to\":[
\"41793026727\",\"41793026834\"],\"text\":\"Test  SMS.\"}",
ParameterType.RequestBody);

IRestResponse response = client.Execute(request);
```

JavaScript

```
var data = JSON.stringify({
  "from": "InfoSMS",
  "to": [
    "41793026727",
    "41793026834"
  ],
  "text": "Test SMS."
});

var xhr = new XMLHttpRequest();
xhr.withCredentials = true;

xhr.addEventListener("readystatechange", function () {
  if (this.readyState === this.DONE) {
    console.log(this.responseText);
  }
});

xhr.open("POST", "http://107.20.199.106/restapi/sms/1/text/single");
xhr.setRequestHeader("authorization", "Basic
QWxhZGRpbjpvcGVuIHNlc2FtZQ==");
xhr.setRequestHeader("content-type", "application/json");
xhr.setRequestHeader("accept",  "application/json");

xhr.send(data);
```

Response

JSON

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

{
   "bulkId":"f5c4322c-10e7-a41e-5528-34fa0b032134",
   "messages":[
      {
         "to":"41793026727",
         "status":{
            "groupId":0,
            "groupName":"ACCEPTED",
            "id":0,
            "name":"MESSAGE_ACCEPTED",
            "description":"Message accepted"
         },
         "smsCount":1,
         "messageId":"4a54f0242f19-b832-1c39-a7e7a2095f351ed2"
      },
      {
         "to":"41793026834",
         "status":{
            "groupId":0,
            "groupName":"ACCEPTED",
            "id":0,
            "name":"MESSAGE_ACCEPTED",
            "description":"Message accepted"
         },
         "smsCount":1,
         "messageId":"9404a69cef19-7a31-ba39-92ace76a5f351ed2"
      }
   ]
}
```

XML

```
HTTP/1.1 200 OK
Content-Type: application/xml

<SMSResponse>
   <bulkId>f5c4322c-10e7-a41e-5528-34fa0b032134</bulkId>
   <messages>
```

```
        <messages>
           <to>41793026727</to>
           <status>
              <id>0</id>
              <groupId>0</groupId>
              <groupName>ACCEPTED</groupName>
              <name>MESSAGE_ACCEPTED</name>
              <description>Message  accepted</description>
           </status>
           <smsCount>1</smsCount>
           <messageId>2250be2d4219-3af1-78856-aabe-1362af1edfd2</messageId>
        </messages>
        <messages>
           <to>41793026834</to>
           <status>
              <id>0</id>
              <groupId>0</groupId>
              <groupName>ACCEPTED</groupName>
              <name>MESSAGE_ACCEPTED</name>
              <description>Message  accepted</description>
           </status>
           <smsCount>1</smsCount>
           <messageId>9404a69cef19-7a31-ba39-92ace76a5f351ed2</messageId>
        </messages>
     </messages>
  </SMSResponse>
```

Multiple textual message

This method allows you to send multiple textual messages to one or more destination addresses.

Definition

`http://107.20.199.106/restapi/sms/1/text/multi`

Parameters

Parameter	Type	Description
messages	SMSTextualData[]	Messages that you want to send.

SMSMultiTextualData

Parameter	Type	Description
from	String	Represents sender ID and it can be alphanumeric or numeric. *Alphanumeric* sender ID length should be between 3 and 11 characters (Example: `CompanyName`). *Numeric* sender ID length should be between 3 and 14 characters.
to	String[]	**Required.** Array of message destination addresses. If you want to send a message to one destination, a single String is supported instead of an Array. Destination addresses must be in international format (Example: `41793026727`).
text	String	Text of the message that will be sent.

Examples

```
POST /restapi/sms/1/text/multi HTTP/1.1
Host: 107.20.199.106
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
Content-Type: application/json
Accept: application/json

{
   "messages":[
      {
         "from":"InfoSMS",
         "to":[
            "41793026727",
            "41793026731"
         ],
         "text":"May the Force be with you!"
      },
      {
         "from":"41793026700",
         "to":"41793026785",
         "text":"A long time ago, in a galaxy far, far away... It is a period of civil war. Rebel spaceships, striking from a hidden base, have won their first victory against the evil Galactic Empire."
      }
   ]
}
```

Result Format


```
{
   "bulkId":"5028e2d42f19-42f1-4656-351e-a42c191e5fd2",
   "messages":[
      {
         "to":"41793026727",
         "status":{
            "groupId":0,
            "groupName":"ACCEPTED",
            "id":0,
            "name":"MESSAGE_ACCEPTED",
            "description":"Message accepted"
         },
         "smsCount":1,
         "messageId":"4242f196ba50-a356-2f91-831c4aa55f351ed2"
      },
      {
         "to":"41793026785",
         "status":{
            "groupId":0,
            "groupName":"ACCEPTED",
            "id":0,
            "name":"MESSAGE_ACCEPTED",
            "description":"Message accepted"
         },
         "smsCount":2,
         "messageId":"5f35f87a2f19-a141-43a4-91cd81b85f8c689"
      }
   ]
}
```


Responseformat

If successful, response header HTTP status code will be `200 OK` and the message will be sent.

If you try to send message without authorization, you will receive an error `401 Unauthorized`.

SMSResponse

Parameter	Type	Description

bulkId	String	The ID that uniquely identifies the request. Bulk ID will be received only when you send a message to more than one destination address.
messages	SMSResponseDetails[]	Array of sent message objects, one object per every message.

SMSResponseDetails

Parameter	Type	Description
to	String	The message destination address.
status	Status	Indicates whether the message is successfully sent, not sent, delivered, not delivered, waiting for delivery or any other possible status.
smsCount	int	The number of sent message segments.
messageId	String	The ID that uniquely identifies the message sent.

Status

Parameter	Type	Description
groupId	int	Status group ID.
groupName	String	Status group name.
id	int	Status ID.
name	String	Status name.
description	String	Human readable description of the status.
action	String	Action that should be taken to eliminate the error.

> **Long SMS:**
>
> Maximum length for one message is 160 characters for GSM7 standard or 70 characters Unicode encoded messages. If you send text which exceeds the maximum number of supported characters for one message, the sent message will be segmented and charged accordingly. One Long SMS that consists of two SMS counts as two SMS.

Additional examples

Multiple textual messages to multiple destinations

Request:

JSON

```
POST /restapi/sms/1/text/multi HTTP/1.1
Host: 107.20.199.106
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
Content-Type: application/json
Accept: application/json

{
   "messages":[
      {
         "from":"InfoSMS",
         "to":[
            "41793026727",
            "41793026731"
         ],
         "text":"May the Force be with you!"
      },
      {
         "from":"41793026700",
         "to":"41793026785",
         "text":"A long time ago, in a galaxy far, far away... It is a period of civil war. Rebel spaceships, striking from a hidden base, have won their first victory against the evil Galactic Empire."
      }
   ]
}
```

XML

```
POST /restapi/sms/1/text/multi HTTP/1.1
Host: 107.20.199.106
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
Content-Type: application/xml
Accept: application/xml

<request>
   <messages>
      <messages>
```

```
            <from>InfoSMS</from>
            <to>
               <to>41793026727</to>
               <to>41793026731</to>
            </to>
            <text>May the Force be with you!</text>
         </messages>
         <messages>
            <from>41793026700</from>
            <to>
               <to>41793026785</to>
            </to>
            <text>A long time ago, in a galaxy far, far away... It is a period
of civil war. Rebel spaceships, striking from a hidden base, have won their
first victory against the evil Galactic Empire.</text>
         </messages>
      </messages>
</request>
```

cURL

```
curl -X POST \
 -H 'Content-Type: application/json' \
 -H 'Accept: application/json' \
 -H 'Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==' \
 -d '{
   "messages":[
      {
         "from":"InfoSMS",
         "to":[
            "41793026727",
            "41793026731"
         ],
         "text":"May the Force be with you!"
      },
      {
         "from":"41793026700",
         "to":"41793026785",
         "text":"A long time ago, in a galaxy far, far away... It is a
period of civil war. Rebel spaceships, striking from a hidden base, have
won their first victory against the evil Galactic Empire."
      }
   ]
}'  http://107.20.199.106/restapi/sms/1/text/multi
```

PHP

```php
<?php

$request = new HttpRequest();
$request->setUrl('http://107.20.199.106/restapi/sms/1/text/multi');
$request->setMethod(HTTP_METH_POST);

$request->setHeaders(array(
  'authorization' => 'Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==',
  'content-type' => 'application/json'
));

$request->setBody('{
   "messages":[
      {
         "from":"InfoSMS",
         "to":[
            "41793026727",
            "41793026731"
         ],
         "text":"May the Force be with you!"
      },
      {
         "from":"41793026700",
         "to":"41793026785",
         "text":"A long time ago, in a galaxy far, far away... It is a
period of civil war. Rebel spaceships, striking from a hidden base, have
won their first victory against the evil Galactic Empire."
      }
   ]
}');

try {
  $response = $request->send();

  echo $response->getBody();
} catch (HttpException $ex) {
  echo $ex;
}
```

Ruby

```
require 'uri'
require 'net/http'

url = URI("http://107.20.199.106/restapi/sms/1/text/multi")

http = Net::HTTP.new(url.host, url.port)
http.use_ssl = true
http.verify_mode = OpenSSL::SSL::VERIFY_NONE

request = Net::HTTP::Post.new(url)
request["authorization"] = 'Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ=='
request["content-type"] = 'application/json'
request["accept"] = 'application/json'

request.body = "{\"messages\":[{\"from\":\"InfoSMS\",\"to\":
[\"41793026727\", \"41793026731\"], \"text\":\"May the Force be with
you!\"}, {\"from\":\"41793026700\",\"to\":\"41793026785\", \"text\":\"A
long time ago, in a galaxy far, far away... It is a period of civil war.
Rebel spaceships, striking from a hidden base, have won their first victory
against the evil Galactic Empire.\"}]}"

response = http.request(request)
puts response.read_body
```

Python

```
import http.client

conn = http.client.HTTPConnection("107.20.199.106")

payload = "{\"messages\":[{\"from\":\"InfoSMS\", \"to\":[\"41793026727\",
\"41793026731\"], \"text\":\"May the Force be with you!\"},
{\"from\":\"41793026700\", \"to\":\"41793026785\", \"text\":\"A long time
ago, in a galaxy far, far away... It is a period of civil war. Rebel
spaceships, striking from a hidden base, have won their first victory
against the evil Galactic Empire.\"}]}"

headers = {
    'authorization': "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==",
    'content-type':  "application/json",
    'accept': "application/json"
    }
```

```
conn.request("POST", "/restapi/sms/1/text/multi", payload, headers)

res = conn.getresponse()
data = res.read()

print(data.decode("utf-8"))
```

Java

```
HttpResponse<String> response =
Unirest.post("http://107.20.199.106/restapi/sms/1/text/multi")
  .header("authorization", "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==")
  .header("content-type",  "application/json")
  .header("accept",  "application/json")
  .body("{\"messages\":[{\"from\":\"InfoSMS\",   \"to\":[\"41793026727\",
\"41793026731\"], \"text\":\"May the Force be with you!\"},
{\"from\":\"41793026700\", \"to\":\"41793026785\", \"text\":\"A long time
ago, in a galaxy far, far away... It is a period of civil war. Rebel
spaceships, striking from a hidden base, have won their first victory
against the evil Galactic Empire.\"}]}")
  .asString();
```

C#

```
var client = new
RestClient("http://107.20.199.106/restapi/sms/1/text/multi");

var request = new RestRequest(Method.POST);
request.AddHeader("accept", "application/json");
request.AddHeader("content-type",   "application/json");
request.AddHeader("authorization", "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==");
request.AddParameter("application/json", "{\"messages\":
[{\"from\":\"InfoSMS\",  \"to\":[\"41793026727\",  \"41793026731\"],
\"text\":\"May the Force be with you!\"}, {\"from\":\"41793026700\",
\"to\":\"41793026785\", \"text\":\"A long time ago, in a galaxy far, far
away... It is a period of civil war. Rebel spaceships, striking from a
hidden base, have won their first victory against the evil Galactic
Empire.\"}]}",  ParameterType.RequestBody);

IRestResponse response = client.Execute(request);
```

JavaScript

```
var data = JSON.stringify({
  "messages": [
    {
      "from": "InfoSMS",
      "to": [
        "41793026727",
        "41793026731"
      ],
      "text": "May the Force be with you!"
    },
    {
      "from": "41793026700",
      "to": "41793026785",
      "text": "A long time ago, in a galaxy far, far away... It is a period
of civil war. Rebel spaceships, striking from a hidden base, have won their
first victory against the evil Galactic Empire."
    }
  ]
});

var xhr = new XMLHttpRequest();
xhr.withCredentials = true;

xhr.addEventListener("readystatechange", function () {
  if (this.readyState === this.DONE) {
    console.log(this.responseText);
  }
});

xhr.open("POST", "http://107.20.199.106/restapi/sms/1/text/multi");
xhr.setRequestHeader("authorization", "Basic
QWxhZGRpbjpvcGVuIHNlc2FtZQ==");
xhr.setRequestHeader("content-type", "application/json");
xhr.setRequestHeader("accept",  "application/json");

xhr.send(data);
```

Response

JSON

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

```
{
   "bulkId":"5028e2d42f19-42f1-4656-351e-a42c191e5fd2",
   "messages":[
      {
         "to":"41793026727",
         "status":{
            "groupId":0,
            "groupName":"ACCEPTED",
            "id":0,
            "name":"MESSAGE_ACCEPTED",
            "description":"Message accepted"
         },
         "smsCount":1,
         "messageId":"4242f196ba50-a356-2f91-831c4aa55f351ed2"
      },
      {
         "to":"41793026731",
         "status":{
            "groupId":0,
            "groupName":"ACCEPTED",
            "id":0,
            "name":"MESSAGE_ACCEPTED",
            "description":"Message accepted"
         },
         "smsCount":1,
         "messageId":"9304a5a3ab19-1ca1-be74-76ad87651ed25f35"
      },
      {
         "to":"41793026785",
         "status":{
            "groupId":0,
            "groupName":"ACCEPTED",
            "id":0,
            "name":"MESSAGE_ACCEPTED",
            "description":"Message accepted"
         },
         "smsCount":2,
         "messageId":"5f35f87a2f19-a141-43a4-91cd81b85f8c689"
      }
   ]
}
```


XML

```
HTTP/1.1 200 OK
Content-Type: application/xml

<SendSMSResponse>
   <bulkId>7241ea9b2ca9-a11f-42c6-aa1e-83cd11c75f52</bulkId>
   <messages>
      <messages>
         <to>41793026727</to>
         <status>
            <groupId>0</groupId>
            <groupName>ACCEPTED</groupName>
            <id>0</id>
            <name>MESSAGE_ACCEPTED</name>
            <description>Message  accepted</description>
         </status>
         <smsCount>1</smsCount>
         <messageId>4242f196ba50-a356-2f91-831c4aa55f351ed2</messageId>
      </messages>
      <messages>
         <to>41793026731</to>
         <status>
            <groupId>0</groupId>
            <groupName>ACCEPTED</groupName>
            <id>0</id>
            <name>MESSAGE_ACCEPTED</name>
            <description>Message  accepted</description>
         </status>
         <smsCount>1</smsCount>
         <messageId>9304a5a3ab19-1ca1-be74-76ad87651ed25f35</messageId>
      </messages>
      <messages>
         <to>41793026785</to>
         <status>
            <groupId>0</groupId>
            <groupName>ACCEPTED</groupName>
            <id>0</id>
            <name>MESSAGE_ACCEPTED</name>
            <description>Message  accepted</description>
         </status>
         <smsCount>2</smsCount>
         <messageId>5f35f87a2f19-a141-43a4-91cd81b85f8c689</messageId>
      </messages>
   </messages>
</SendSMSResponse>
```

Single binary message

This method allows you to send single binary message to one or more destination addresses.

Definition

`http://107.20.199.106/restapi/sms/1/binary/single`

Parameters

Parameter	Type	Description
from	String	Represents sender ID and it can be alphanumeric or numeric. *Alphanumeric* sender ID length should be between 3 and 11 characters (Example: `CompanyName`). *Numeric* sender ID length should be between 3 and 14 characters.
to	String[]	**Required.** Array of message destination addresses. If you want to send a message to one destination, a single String is supported instead of an Array. Destination addresses must be in international format (Example: `41793026727`).
binary	BinaryContent	Binary content that you want to send.

BinaryContent

Parameter	Type	Description
hex	String	Hexadecimal string. This is the representation of your binary data. Two hex digits represent one byte. They should be separated by space character (Example: `"0f c2 4a bf 34 13 ba"`).
dataCoding	int	Binary content data coding. Default value is (`0`) for GSM7. Example: (`8`) for Unicode data.
esmClass	int	“Esm_class” parameter. Indicate a special message attributes associated with the SMS. Default value is `(0)`.

> Find out more about Data coding and “Esm_class” usage here.

Examples

```
POST /1/sms/binary/single HTTP/1.1
Host: 107.20.199.106
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
Content-Type: application/json
Accept: application/xml

{
   "from":"InfoSMS",
   "to":"41793026727",
   "binary":{
      "hex":"54 65 73 74 20 6d 65 73 73 61 67 65 2e",
      "dataCoding":0,
      "esmClass":0
   }
}
```

Result Format

```
{
   "messages":[
      {
         "to":"41793026727",
         "status":{
            "groupId":0,
            "groupName":"ACCEPTED",
            "id":0,
            "name":"MESSAGE_ACCEPTED",
            "description":"Message accepted"
         },
         "smsCount":1,
         "messageId":"502baa42f19-42f1-351e-4656a42c1b1e5fd2"
      }
   ]
}
```

Responseformat

If successful, response header HTTP status code will be `200 OK` and the message will be sent.

If you try to send message without authorization, you will receive an error `401 Unauthorized`.

SMSResponse

Parameter	Type	Description
bulkId	String	The ID that uniquely identifies the request. Bulk ID will be received only when you send a message to more than one destination address.
messages	SMSResponseDetails[]	Array of sent message objects, one object per every message.

SMSResponseDetails

Parameter	Type	Description
to	String	The message destination address.
status	Status	Indicates whether the message is successfully sent, not sent, delivered, not delivered, waiting for delivery or any other possible status.
smsCount	int	The number of sent message segments.
messageId	String	The ID that uniquely identifies the message sent.

Status

Parameter	Type	Description
groupId	int	Status group ID.
groupName	String	Status group name.
id	int	Status ID.
name	String	Status name.
description	String	Human readable description of the status.
action	String	Action that should be taken to eliminate the error.

> **Note:**
>
> Find out more about Data coding and "Esm_class" usage here.

Additional examples

Single binary message to one destination

Request

JSON

```
POST /1/sms/binary/single HTTP/1.1
Host: 107.20.199.106
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
Content-Type: application/json
Accept: application/xml

{
   "from":"InfoSMS",
   "to":"41793026727",
   "binary":{
      "hex":"54 65 73 74 20 6d 65 73 73 61 67 65 2e",
      "dataCoding":0,
      "esmClass":0
   }
}
```

XML

```
POST /1/sms/binary/single HTTP/1.1
Host: 107.20.199.106
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
Content-Type: application/xml
Accept: application/xml

<request>
   <from>InfoSMS</from>
   <to>
      <to>41793026727</to>
   </to>
   <binary>
      <hex>54 65 73 74 20 6d 65 73 73 61 67 65 2e</hex>
      <dataCoding>0</dataCoding>
      <esmClass>0</esmClass>
   </binary>
```

```
</request>
```

cURL

```
curl -X POST \
 -H 'Content-Type: application/json' \
 -H 'Accept: application/json' \
 -H 'Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==' \
 -d '{
   "from":"InfoSMS",
   "to":"41793026727",
   "binary":{
      "hex":"54 65 73 74 20 6d 65 73 73 61 67 65 2e",
      "dataCoding":0,
      "esmClass":0
   }
}'  http://107.20.199.106/restapi/1/sms/binary/single
```

PHP

```
<?php

$request = new HttpRequest();
$request->setUrl('http://107.20.199.106/restapi/sms/1/binary/single');
$request->setMethod(HTTP_METH_POST);

$request->setHeaders(array(
  'accept' => 'application/json',
  'content-type' => 'application/json',
  'authorization' => 'Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ=='
));

$request->setBody('{
   "from":"InfoSMS",
   "to":"41793026727",
   "binary":{
      "hex":"54 65 73 74 20 6d 65 73 73 61 67 65 2e",
      "dataCoding":0,
      "esmClass":0
   }
}');

try {
```

```
  $response = $request->send();

  echo $response->getBody();
} catch (HttpException $ex) {
  echo $ex;
}
```

Ruby

```
require 'uri'
require 'net/http'

url = URI("http://107.20.199.106/restapi/sms/1/binary/single")

http = Net::HTTP.new(url.host, url.port)
http.use_ssl = true
http.verify_mode = OpenSSL::SSL::VERIFY_NONE

request = Net::HTTP::Post.new(url)
request["authorization"] = 'Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ=='
request["content-type"] = 'application/json'
request["accept"] = 'application/json'

request.body = "{\"from\":\"InfoSMS\", \"to\":\"41793026727\", \"binary\":
{\"hex\":\"54 65 73 74 20 6d 65 73 73 61 67 65 2e\", \"dataCoding\":0,
\"esmClass\":0}}"

response = http.request(request)
puts response.read_body
```

Python

```
import http.client

conn = http.client.httpConnection("107.20.199.106")

payload = "{\"from\":\"InfoSMS\", \"to\":\"41793026727\", \"binary\":
{\"hex\":\"54 65 73 74 20 6d 65 73 73 61 67 65 2e\", \"dataCoding\":0,
\"esmClass\":0}}"

headers = {
    'authorization': "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==",
    'content-type':  "application/json",
```

```
    'accept': "application/json"
    }

conn.request("POST", "/restapi/sms/1/binary/single", payload, headers)

res = conn.getresponse()
data = res.read()

print(data.decode("utf-8"))
```

Java

```
HttpResponse<String> response =
Unirest.post("http://107.20.199.106/restapi/sms/1/binary/single")
  .header("authorization", "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==")
  .header("content-type",  "application/json")
  .header("accept",  "application/json")
  .body("{\"from\":\"InfoSMS\",  \"to\":\"41793026727\",  \"binary\":
{\"hex\":\"54 65 73 74 20 6d 65 73 73 61 67 65 2e\", \"dataCoding\":0,
\"esmClass\":0}}")
  .asString();
```

C#

```
var client = new
RestClient("http://107.20.199.106/restapi/sms/1/binary/single");

var request = new RestRequest(Method.POST);
request.AddHeader("accept", "application/json");
request.AddHeader("content-type",   "application/json");
request.AddHeader("authorization",  "Basic  QWxhZGRpbjpvcGVuIHNlc2FtZQ==");
request.AddParameter("application/json",   "{\"from\":\"InfoSMS\",
\"to\":\"41793026727\", \"binary\":{\"hex\":\"54 65 73 74 20 6d 65 73 73 61
67 65 2e\", \"dataCoding\":0, \"esmClass\":0}}",
ParameterType.RequestBody);

IRestResponse response = client.Execute(request);
```

JavaScript

```
var data = JSON.stringify({
  "from": "InfoSMS",
```

```
  "to": "41793026727",
  "binary": {
    "hex": "54 65 73 74 20 6d 65 73 73 61 67 65 2e",
    "dataCoding": 0,
    "esmClass": 0
  }
});

var xhr = new XMLHttpRequest();
xhr.withCredentials = true;

xhr.addEventListener("readystatechange", function () {
  if (this.readyState === this.DONE) {
    console.log(this.responseText);
  }
});

xhr.open("POST", "http://107.20.199.106/restapi/sms/1/binary/single");
xhr.setRequestHeader("authorization", "Basic
QWxhZGRpbjpvcGVuIHNlc2FtZQ==");
xhr.setRequestHeader("content-type", "application/json");
xhr.setRequestHeader("accept",  "application/json");

xhr.send(data);
```

Response

JSON

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

{
   "messages":[
      {
         "to":"41793026727",
         "status":{
            "groupId":0,
            "groupName":"ACCEPTED",
            "id":0,
            "name":"MESSAGE_ACCEPTED",
            "description":"Message accepted"
         },
         "smsCount":1,
```

```
            "messageId":"502baa42f19-42f1-351e-4656a42c1b1e5fd2"
         }
      ]
   }
```

XML

```
HTTP/1.1 200 OK
Content-Type: application/xml

<SMSResponse>
   <messages>
      <messages>
         <to>41793026727</to>
         <status>
            <groupId>0</groupId>
            <groupName>ACCEPTED</groupName>
            <id>0</id>
            <name>MESSAGE_ACCEPTED</name>
            <description>Message  accepted</description>
         </status>
         <smsCount>1</smsCount>
         <messageId>502baa42f19-42f1-351e-4656a42c1b1e5fd2</messageId>
      </messages>
   </messages>
</SMSResponse>
```

Single binary message to multiple destinations

Request

JSON

```
POST /1/sms/binary/single HTTP/1.1
Host: 107.20.199.106
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
Content-Type: application/json
Accept: application/json

{
   "from":"InfoSMS",
   "to":[
```

```
        "41793026727",
        "41793026834"
    ],
    "binary":{
        "hex":"54 65 73 74 69 6e 67 20 c4 8d c5 a1 c4 87 c4 91",
        "dataCoding":7,
        "esmClass":0
    }
}
```

XML

```
POST /1/sms/binary/single HTTP/1.1
Host: 107.20.199.106
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
Content-Type: application/xml
Accept: application/xml

<request>
    <from>InfoSMS</from>
    <to>
        <to>41793026727</to>
        <to>41793026834</to>
    </to>
    <binary>
        <hex>54 65 73 74 69 6e 67 20 c4 8d c5 a1 c4 87 c4 91</hex>
        <dataCoding>7</dataCoding>
        <esmClass>0</esmClass>
    </binary>
</request>
```

cURL

```
curl -X POST
 -H 'Content-Type: application/json'
 -H 'Accept: application/json'
 -H 'Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ=='
 -d '{
   "from":"InfoSMS",
   "to":[
      "41793026727",
      "41793026834"
   ],
```

```
    "binary":{
       "hex":"54 65 73 74 69 6e 67 20 c4 8d c5 a1 c4 87 c4 91",
       "dataCoding":7,
       "esmClass":0
    }
}'  http://107.20.199.106/restapi/1/sms/binary/single
```

PHP

```
<?php

$request = new HttpRequest();
$request->setUrl('http://107.20.199.106/restapi/sms/1/binary/single');
$request->setMethod(HTTP_METH_POST);

$request->setHeaders(array(
  'accept' => 'application/json',
  'content-type' => 'application/json',
  'authorization' => 'Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ=='
));

$request->setBody('{
   "from":"InfoSMS",
   "to":[
      "41793026727",
      "41793026834"
   ],
   "binary":{
      "hex":"54 65 73 74 69 6e 67 20 c4 8d c5 a1 c4 87 c4 91",
      "dataCoding":7,
      "esmClass":0
   }
}');

try {
  $response = $request->send();

  echo $response->getBody();
} catch (HttpException $ex) {
  echo $ex;
}
```

Ruby

```ruby
require 'uri'
require 'net/http'

url = URI("http://107.20.199.106/restapi/sms/1/binary/single")

http = Net::HTTP.new(url.host, url.port)
http.use_ssl = true
http.verify_mode = OpenSSL::SSL::VERIFY_NONE

request = Net::HTTP::Post.new(url)
request["authorization"] = 'Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ=='
request["content-type"] = 'application/json'
request["accept"] = 'application/json'

request.body = "{\"from\":\"InfoSMS\", \"to\":[\"41793026727\",
\"41793026834\"], \"binary\":{\"hex\":\"54 65 73 74 69 6e 67 20 c4 8d c5 a1
c4 87 c4 91\", \"dataCoding\":7, \"esmClass\":0}}"

response = http.request(request)
puts response.read_body
```

Python

```python
import http.client

conn = http.client.httpConnection("107.20.199.106")

payload = "{\"from\":\"InfoSMS\", \"to\":[\"41793026727\",
\"41793026834\"], \"binary\":{\"hex\":\"54 65 73 74 69 6e 67 20 c4 8d c5 a1
c4 87 c4 91\", \"dataCoding\":7, \"esmClass\":0}}"

headers = {
    'authorization': "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==",
    'content-type':  "application/json",
    'accept': "application/json"
    }

conn.request("POST", "/restapi/sms/1/binary/single", payload, headers)

res = conn.getresponse()
data = res.read()

print(data.decode("utf-8"))
```

Java

```
HttpResponse<String> response =
Unirest.post("http://107.20.199.106/restapi/sms/1/binary/single")
  .header("authorization", "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==")
  .header("content-type",  "application/json")
  .header("accept",  "application/json")
  .body("{\"from\":\"InfoSMS\",  \"to\":[\"41793026727\",  \"41793026834\"],
\"binary\":{\"hex\":\"54 65 73 74 69 6e 67 20 c4 8d c5 a1 c4 87 c4 91\",
\"dataCoding\":7,  \"esmClass\":0}}")
  .asString();
```

C#

```
var client = new
RestClient("http://107.20.199.106/restapi/sms/1/binary/single");

var request = new RestRequest(Method.POST);
request.AddHeader("accept", "application/json");
request.AddHeader("content-type",  "application/json");
request.AddHeader("authorization", "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==");
request.AddParameter("application/json", "{\"from\":\"InfoSMS\", \"to\":
[\"41793026727\", \"41793026834\"], \"binary\":{\"hex\":\"54 65 73 74 69 6e
67 20 c4 8d c5 a1 c4 87 c4 91\", \"dataCoding\":7, \"esmClass\":0}}",
ParameterType.RequestBody);

IRestResponse response = client.Execute(request);
```

JavaScript

```
var data = JSON.stringify({
  "from": "InfoSMS",
  "to": [
    "41793026727",
    "41793026834"
  ],
  "binary": {
    "hex": "54 65 73 74 69 6e 67 20 c4 8d c5 a1 c4 87 c4 91",
    "dataCoding": 7,
    "esmClass": 0
  }
});
```

```
var xhr = new XMLHttpRequest();
xhr.withCredentials = true;

xhr.addEventListener("readystatechange", function () {
  if (this.readyState === this.DONE) {
    console.log(this.responseText);
  }
});

xhr.open("POST", "http://107.20.199.106/restapi/sms/1/binary/single");
xhr.setRequestHeader("authorization", "Basic
QWxhZGRpbjpvcGVuIHNlc2FtZQ==");
xhr.setRequestHeader("content-type", "application/json");
xhr.setRequestHeader("accept",  "application/json");

xhr.send(data);
```

Response

JSON

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

{
   "bulkId":"5028e2d42f19-42f1-4656-351e-a42c191e5fd2",
   "messages":[
      {
         "to":"41793026727",
         "status":{
            "groupId":0,
            "groupName":"ACCEPTED",
            "id":0,
            "name":"MESSAGE_ACCEPTED",
            "description":"Message accepted"
         },
         "smsCount":1,
         "messageId":"465650242f19-4656-c191-a42e87655f351ed2"
      },
      {
         "to":"41793026834",
         "status":{
            "groupId":0,
```

```
            "groupName":"ACCEPTED",
            "id":0,
            "name":"MESSAGE_ACCEPTED",
            "description":"Message accepted"
         },
         "smsCount":1,
         "messageId":"1234aba32f19-3c31-b294-76ad87655f351ed2"
      }
   ]
}
```


XML


```
HTTP/1.1 200 OK
Content-Type: application/xml

<SMSResponse>
   <bulkId>7241ea9b2ca9-a11f-42c6-aa1e-83cd11c75f52</bulkId>
   <messages>
      <messages>
         <to>41793026727</to>
         <status>
            <groupId>0</groupId>
            <groupName>ACCEPTED</groupName>
            <id>0</id>
            <name>MESSAGE_ACCEPTED</name>
            <description>Message  accepted</description>
         </status>
         <smsCount>1</smsCount>
         <messageId>4242f196ba50-a356-2f91-831c4aa55f351ed2</messageId>
      </messages>
      <messages>
         <to>41793026834</to>
         <status>
            <groupId>0</groupId>
            <groupName>ACCEPTED</groupName>
            <id>0</id>
            <name>MESSAGE_ACCEPTED</name>
            <description>Message  accepted</description>
         </status>
         <smsCount>1</smsCount>
         <messageId>9304a5a3ab19-1ca1-be74-76ad87651ed25f35</messageId>
      </messages>
   </messages>
```

```
</SMSResponse>
```

Multiple binary message

This method allows you to send multiple binary messages to one or more destination addresses.

Definition

```
http://107.20.199.106/restapi/sms/1/binary/multi
```

Parameters

Parameter	Type	Description
messages	SMSBinaryData[]	Messages that you want to send.

SMSBinaryData

Parameter	Type	Description
from	String	Represents sender ID and it can be alphanumeric or numeric. *Alphanumeric* sender ID length should be between 3 and 11 characters (Example: `CompanyName`). *Numeric* sender ID length should be between 3 and 14 characters.
to	String[]	**Required.** Array of message destination addresses. If you want to send a message to one destination, a single String is supported instead of an Array. Destination addresses must be in international format (Example: `41793026727`).
binary	BinaryContent	Binary content that you want to send.

BinaryContent

Parameter	Type	Description
hex	String	Hexadecimal string. This is the representation of your binary data. Two hex digits represent one byte. They should be separated by space character (Example: `"0f c2 4a bf 34 13 ba"`).
dataCoding	Integer	

		Binary content data coding. Default value is (`0`) for GSM7. Example: (`8`) for Unicode data.
esmClass	Integer	“Esm_class” parameter. Indicate a special message attributes associated with the SMS. Default value is `(0)`.

Examples

```
POST /1/sms/binary/multi HTTP/1.1
Host: 107.20.199.106
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
Content-Type: application/json
Accept: application/json

{
   "messages":[
      {
         "from":"InfoSMS",
         "to":[
            "41793026727",
            "41793026731"
         ],
         "binary":{
            "hex":"54 65 73 74 20 6d 65 73 73 61 67 65 2e",
            "dataCoding":0,
            "esmClass":0
         }
      },
      {
         "from":"41793026700",
         "to":"41793026785",
         "binary":{
            "hex":"d1 82 d0 b5 d1 81 d1 82 d0 b8 d1 80 d0 b0 d1 9a d0 b5 20
d1 9b d0 b8 d1 80 d0 b8 d0 bb d0 b8 d1 86 d0 b5",
            "dataCoding":6,
            "esmClass":0
         }
      }
   ]
}
```

Result Format


```
{
   "bulkId":"1036A4cb-803a-4a73-a5a6-7e4fc3791722",
   "messages":[
      {
         "to":"41793026727",
         "status":{
            "groupId":7,
            "groupName":"PENDING",
            "id":7,
            "name":"PENDING_ENROUTE",
            "description":"Message sent to next instance"
         },
         "smsCount":1,
         "messageId":"1254df3e-ab62-15af5-230b-14d8e65c7540"
      },
      {
         "to":"41793026785",
         "status":{
            "groupId":1,
            "groupName":"PENDING",
            "id":7,
            "name":"PENDING_ENROUTE",
            "description":"Message sent to next instance"
         },
         "smsCount":1,
         "messageId":"1877df3e-4b62-46f5-819b-14d8e65c2291"
      }
   ]
}
```


Responseformat

If successful, response header HTTP status code will be `200 OK` and the messages will be sent.

If you try to send messages without authorization, you will receive an error `401 Unauthorized`.

SMSResponse

Parameter	Type	Description

bulkId	String	The ID that uniquely identifies the request. Bulk ID will be received only when you send the message to more than one destination address.
messages	SMSResponseDetails[]	Array of sent message objects, one object per every message.

SMSResponseDetails

Parameter	Type	Description
to	String	The message destination address.
status	Status	Indicates whether the message is successfully sent, not sent, delivered, not delivered, waiting for delivery or any other possible status.
smsCount	int	The number of sent message segments.
messageId	String	The ID that uniquely identifies the message sent.

Status

Parameter	Type	Description
groupId	int	Status group ID.
groupName	String	Status group name.
id	int	Status ID.
name	String	Status name.
description	String	Human readable description of the status.
action	String	Action that should be taken to eliminate the error.

> **Note:**
>
> Find out more about Data coding and Esm_class usage here.

Additional examples

Multiple binary messages to multiple destinations

Request

JSON

```
POST /1/sms/binary/multi HTTP/1.1
Host: 107.20.199.106
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
Content-Type: application/json
Accept: application/json

{
   "messages":[
      {
         "from":"InfoSMS",
         "to":[
            "41793026727",
            "41793026731"
         ],
         "binary":{
            "hex":"54 65 73 74 20 6d 65 73 73 61 67 65 2e",
            "dataCoding":0,
            "esmClass":0
         }
      },
      {
         "from":"41793026700",
         "to":"41793026785",
         "binary":{
            "hex":"d1 82 d0 b5 d1 81 d1 82 d0 b8 d1 80 d0 b0 d1 9a d0 b5 20
d1 9b d0 b8 d1 80 d0 b8 d0 bb d0 b8 d1 86 d0 b5",
            "dataCoding":6,
            "esmClass":0
         }
      }
   ]
}
```

XML

```
POST /1/sms/binary/multi HTTP/1.1
Host: 107.20.199.106
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
Content-Type: application/xml
Accept: application/xml
```

```
<request>
   <messages>
      <messages>
         <from>InfoSMS</from>
         <to>
            <to>41793026727</to>
            <to>41793026731</to>
         </to>
         <binary>
            <hex>54 65 73 74 20 6d 65 73 73 61 67 65 2e</hex>
            <dataCoding>0</dataCoding>
            <esmClass>0</esmClass>
         </binary>
      </messages>
      <messages>
         <from>41793026700</from>
         <to>
            <to>41793026785</to>
         </to>
         <binary>
            <hex>d1 82 d0 b5 d1 81 d1 82 d0 b8 d1 80 d0 b0 d1 9a d0 b5 20
d1 9b d0 b8 d1 80 d0 b8 d0 bb d0 b8 d1 86 d0 b5</hex>
            <dataCoding>6</dataCoding>
            <esmClass>0</esmClass>
         </binary>
      </messages>
   </messages>
</request>
```

cURL

```
curl -X POST \
-H 'Content-Type: application/json' \
-H 'Accept: application/json'\
-H 'Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==' \
-d '{
   "messages":[
      {
         "from":"InfoSMS",
         "to":[
            "41793026727",
            "41793026731"
         ],
```

```
            "binary":{
               "hex":"54 65 73 74 20 6d 65 73 73 61 67 65 2e",
               "dataCoding":0,
               "esmClass":0
            }
         },
         {
            "from":"41793026700",
            "to":"41793026785",
            "binary":{
               "hex":"d1 82 d0 b5 d1 81 d1 82 d0 b8 d1 80 d0 b0 d1 9a d0 b5 20
d1 9b d0 b8 d1 80 d0 b8 d0 bb d0 b8 d1 86 d0 b5",
               "dataCoding":6,
               "esmClass":0
            }
         }
      ]
   }'  http://107.20.199.106/restapi/1/sms/binary/multi
```

PHP

```
<?php

$request = new HttpRequest();
$request->setUrl('http://107.20.199.106/restapi/sms/1/binary/multi');
$request->setMethod(HTTP_METH_POST);

$request->setHeaders(array(
  'accept' => 'application/json',
  'content-type' => 'application/json',
  'authorization' => 'Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ=='
));

$request->setBody('{
   "messages":[
      {
         "from":"InfoSMS",
         "to":[
            "41793026727",
            "41793026731"
         ],
         "binary":{
            "hex":"54 65 73 74 20 6d 65 73 73 61 67 65 2e",
            "dataCoding":0,
```

```
            "esmClass":0
         }
      },
      {
         "from":"41793026700",
         "to":"41793026785",
         "binary":{
            "hex":"d1 82 d0 b5 d1 81 d1 82 d0 b8 d1 80 d0 b0 d1 9a d0 b5 20
d1 9b d0 b8 d1 80 d0 b8 d0 bb d0 b8 d1 86 d0 b5",
            "dataCoding":6,
            "esmClass":0
         }
      }
   ]
}');

try {
  $response = $request->send();

  echo $response->getBody();
} catch (HttpException $ex) {
  echo $ex;
}
```

Ruby

```
require 'uri'
require 'net/http'

url = URI("http://107.20.199.106/restapi/sms/1/binary/multi")

http = Net::HTTP.new(url.host, url.port)
http.use_ssl = true
http.verify_mode = OpenSSL::SSL::VERIFY_NONE

request = Net::HTTP::Post.new(url)
request["authorization"] = 'Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ=='
request["content-type"] = 'application/json'
request["accept"] = 'application/json'

request.body = "{\"messages\":[{\"from\":\"InfoSMS\", \"to\":
[\"41793026727\", \"41793026731\"], \"binary\":{\"hex\":\"54 65 73 74 20 6d
65 73 73 61 67 65 2e\", \"dataCoding\":0, \"esmClass\":0}},
{\"from\":\"41793026700\",  \"to\":\"41793026785\",  \"binary\":{\"hex\":\"d1
```

```
82 d0 b5 d1 81 d1 82 d0 b8 d1 80 d0 b0 d1 9a d0 b5 20 d1 9b d0 b8 d1 80 d0
b8 d0 bb d0 b8 d1 86 d0 b5\", \"dataCoding\":6, \"esmClass\":0}}]}"

response = http.request(request)
puts response.read_body
```

Python

```
import http.client

conn = http.client.HTTPConnection("107.20.199.106")

payload = "{\"messages\":[{\"from\":\"InfoSMS\", \"to\":[\"41793026727\",
\"41793026731\"], \"binary\":{\"hex\":\"54 65 73 74 20 6d 65 73 73 61 67 65
2e\", \"dataCoding\":0, \"esmClass\":0}}, {\"from\":\"41793026700\",
\"to\":\"41793026785\", \"binary\":{\"hex\":\"d1 82 d0 b5 d1 81 d1 82 d0 b8
d1 80 d0 b0 d1 9a d0 b5 20 d1 9b d0 b8 d1 80 d0 b8 d0 bb d0 b8 d1 86 d0
b5\", \"dataCoding\":6, \"esmClass\":0}}]}"

headers = {
    'authorization': "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==",
    'content-type':  "application/json",
    'accept': "application/json"
    }

conn.request("POST", "/restapi/sms/1/binary/multi", payload, headers)

res = conn.getresponse()
data = res.read()

print(data.decode("utf-8"))
```

Java

```
HttpResponse<String> response =
Unirest.post("http://107.20.199.106/restapi/sms/1/binary/multi")
  .header("authorization", "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==")
  .header("content-type",  "application/json")
  .header("accept",  "application/json")
  .body("{\"messages\":[{\"from\":\"InfoSMS\",   \"to\":[\"41793026727\",
\"41793026731\"], \"binary\":{\"hex\":\"54 65 73 74 20 6d 65 73 73 61 67 65
2e\", \"dataCoding\":0, \"esmClass\":0}}, {\"from\":\"41793026700\",
\"to\":\"41793026785\", \"binary\":{\"hex\":\"d1 82 d0 b5 d1 81 d1 82 d0 b8
```

```
d1 80 d0 b0 d1 9a d0 b5 20 d1 9b d0 b8 d1 80 d0 b8 d0 bb d0 b8 d1 86 d0
b5\", \"dataCoding\":6, \"esmClass\":0}}]}")
  .asString();
```

C#

```
var client = new
RestClient("http://107.20.199.106/restapi/sms/1/binary/multi");

var request = new RestRequest(Method.POST);
request.AddHeader("accept", "application/json");
request.AddHeader("content-type",   "application/json");
request.AddHeader("authorization", "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==");
request.AddParameter("application/json", "{\"messages\":
[{\"from\":\"InfoSMS\",  \"to\":[\"41793026727\",  \"41793026731\"],
\"binary\":{\"hex\":\"54 65 73 74 20 6d 65 73 73 61 67 65 2e\",
\"dataCoding\":0, \"esmClass\":0}}, {\"from\":\"41793026700\",
\"to\":\"41793026785\", \"binary\":{\"hex\":\"d1 82 d0 b5 d1 81 d1 82 d0 b8
d1 80 d0 b0 d1 9a d0 b5 20 d1 9b d0 b8 d1 80 d0 b8 d0 bb d0 b8 d1 86 d0
b5\", \"dataCoding\":6, \"esmClass\":0}}]}", ParameterType.RequestBody);

IRestResponse response = client.Execute(request);
```

JavaScript

```
var data = JSON.stringify({
  "messages": [
    {
      "from": "InfoSMS",
      "to": [
        "41793026727",
        "41793026731"
      ],
      "binary": {
        "hex": "54 65 73 74 20 6d 65 73 73 61 67 65 2e",
        "dataCoding": 0,
        "esmClass": 0
      }
    },
    {
      "from": "41793026700",
      "to": "41793026785",
      "binary": {
```

```
        "hex": "d1 82 d0 b5 d1 81 d1 82 d0 b8 d1 80 d0 b0 d1 9a d0 b5 20 d1
9b d0 b8 d1 80 d0 b8 d0 bb d0 b8 d1 86 d0 b5",
        "dataCoding": 6,
        "esmClass": 0
      }
    }
  ]
});

var xhr = new XMLHttpRequest();
xhr.withCredentials = true;

xhr.addEventListener("readystatechange", function () {
  if (this.readyState === this.DONE) {
    console.log(this.responseText);
  }
});

xhr.open("POST", "http://107.20.199.106/restapi/sms/1/binary/multi");
xhr.setRequestHeader("authorization", "Basic
QWxhZGRpbjpvcGVuIHNlc2FtZQ==");
xhr.setRequestHeader("content-type", "application/json");
xhr.setRequestHeader("accept",  "application/json");

xhr.send(data);
```

Response

JSON

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

{
   "bulkId":"1036A4cb-803a-4a73-a5a6-7e4fc3791722",
   "messages":[
      {
         "to":"41793026727",
         "status":{
            "groupId":7,
            "groupName":"PENDING",
            "id":7,
            "name":"PENDING_ENROUTE",
            "description":"Message sent to next instance"
```

```
            },
            "smsCount":1,
            "messageId":"1254df3e-ab62-15af5-230b-14d8e65c7540"
         },
         {
            "to":"41793026731",
            "status":{
               "groupId":1,
               "groupName":"PENDING",
               "id":7,
               "name":"PENDING_ENROUTE",
               "description":"Message sent to next instance"
            },
            "smsCount":1,
            "messageId":"a5b7df3e-1b62-4af5-840b-1b5de65c2ad7"
         },
         {
            "to":"41793026785",
            "status":{
               "groupId":1,
               "groupName":"PENDING",
               "id":7,
               "name":"PENDING_ENROUTE",
               "description":"Message sent to next instance"
            },
            "smsCount":1,
            "messageId":"1877df3e-4b62-46f5-819b-14d8e65c2291"
         }
      ]
   }
```


XML


```
HTTP/1.1 200 OK
Content-Type: application/xml

<SMSResponse>
   <bulkId>1036A4cb-803a-4a73-a5a6-7e4fc3791722</bulkId>
   <messages>
      <messages>
         <to>41793026727</to>
         <status>
            <groupId>1</groupId>
            <groupName>PENDING</groupName>
```

```
            <id>7</id>
            <name>PENDING_ENROUTE</name>
            <description>Message sent to next instance</description>
         </status>
         <smsCount>1</smsCount>
         <messageId>1254df3e-ab62-15af5-230b-14d8e65c7540</messageId>
      </messages>
      <messages>
         <to>41793026731</to>
         <status>
            <groupId>1</groupId>
            <groupName>PENDING</groupName>
            <id>7</id>
            <name>PENDING_ENROUTE</name>
            <description>Message sent to next instance</description>
         </status>
         <smsCount>1</smsCount>
         <messageId>a5b7df3e-1b62-4af5-840b-1b5de65c2ad7</messageId>
      </messages>
      <messages>
         <to>41793026785</to>
         <status>
            <groupId>1</groupId>
            <groupName>PENDING</groupName>
            <id>7</id>
            <name>PENDING_ENROUTE</name>
            <description>Message sent to next instance</description>
         </status>
         <smsCount>1</smsCount>
         <messageId>1877df3e-4b62-46f5-819b-14d8e65c2291</messageId>
      </messages>
   </messages>
</SMSResponse>
```


Delivery reports

This method allows you to get one time delivery reports for sent SMS.

Definition

```
http://107.20.199.106/restapi/sms/1/reports
```

Parameters

Parameter	Type	Description
bulkId	String	The ID that uniquely identifies the request. Bulk ID will be received only when you send a message to more than one destination address.
messageId	String	The ID that uniquely identifies the message sent.
limit	String	The maximum number of returned delivery reports.

Examples

```
GET /restapi/sms/1/reports HTTP/1.1
Host: 107.20.199.106
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
Accept: application/json
```

Result Format


```
{
   "results":[
      {
         "bulkId":"8c20f086-d82b-48cc-b2b3-3ca5f7aca9fb",
         "messageId":"ff4804ef-6ab6-4abd-984d-ab3b1387e852",
         "to":"385981178",
         "sentAt":"2015-02-12T09:58:20.323+0100",
         "doneAt":"2015-02-12T09:58:20.337+0100",
         "smsCount":1,
         "price":{
            "pricePerMessage":0.01,
            "currency":"EUR"
         },
         "status":{
            "id":5,
            "groupId":3,
            "groupName":"DELIVERED",
            "name":"DELIVERED_TO_HANDSET",
            "description":"Message delivered to handset"
         },
         "error":{
```

```
            "groupId":0,
            "groupName":"OK",
            "id":0,
            "name":"NO_ERROR",
            "description":"No Error",
            "permanent":false
        }
    }
  ]
}
```


Responseformat

On success, response header HTTP status code will be `200 OK` and delivery reports will be returned in the response body.

If you try to send a message without authorization, you will get a response with HTTP status code `401 Unauthorized`.

SMSReportResponse

Parameter	Type	Description
results	SentSMSReport[]	Collection of reports, one per every message.

SentSMSReport

Parameter	Type	Description
bulkId	String	Bulk ID.
messageId	String	Message ID.
to	String	Destination address.
sentAt	Date	Tells when the SMS was sent. Has the following format: `yyyy-MM-dd'T'HH:mm:ss.SSSZ` .
doneAt	Date	Tells when the SMS was finished processing by the platform (ie. delivered to destination, delivered to destination network, etc.)
smsCount	int	Number of parts the sent SMS was split into.

price	Price	Sent SMS price.
status	Status	Indicates whether the message is successfully sent, not sent, delivered, not delivered, waiting for delivery or any other possible status.
error	Error	Indicates whether the error occurred during the query execution.

Price

Parameter	Type	Description
pricePerMessage	BigDecimal	Price per one SMS.
currency	String	The currency in which the price is expressed.

Status

Parameter	Type	Description
groupId	int	Status group ID.
groupName	String	Status group name.
id	int	Status ID.
name	String	Status name.
description	String	Human readable description of the status.
action	String	Action that should be taken to eliminate the error.

Error

Parameter	Type	Description
groupId	int	Error group ID.
groupName	String	Error group name.
id	int	Error ID.
name	String	Error name.
description	String	Human readable description of the error.

permanent	boolean	Tells if the error is permanent.

Additional examples

Getting reports without any query parameter

Request

JSON

```
GET /restapi/sms/1/reports HTTP/1.1
Host: 107.20.199.106
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
Accept: application/json
```

XML

```
GET /restapi/sms/1/reports HTTP/1.1
Host: 107.20.199.106
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
Accept: application/xml
```

cURL

```
curl -X GET \
-H 'Accept: application/json' \
-H "Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==" \
http://107.20.199.106/restapi/sms/1/reports
```

PHP

```
<?php

$request = new HttpRequest();
$request->setUrl('http://107.20.199.106/restapi/sms/1/reports');
$request->setMethod(HTTP_METH_GET);

$request->setHeaders(array(
```

```
  'accept' => 'application/json',
  'authorization' => 'Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ=='
));

try {
  $response = $request->send();

  echo $response->getBody();
} catch (HttpException $ex) {
  echo $ex;
}
```

Ruby

```
require 'uri'
require 'net/http'

url = URI("http://107.20.199.106/restapi/sms/1/reports")

http = Net::HTTP.new(url.host, url.port)
http.use_ssl = true
http.verify_mode = OpenSSL::SSL::VERIFY_NONE

request = Net::HTTP::Get.new(url)
request["authorization"] = 'Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ=='
request["accept"] = 'application/json'

response = http.request(request)
puts response.read_body
```

Python

```
import http.client

conn = http.client.HTTPConnection("107.20.199.106")

headers = {
    'authorization': "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==",
    'accept': "application/json"
    }

conn.request("GET", "/restapi/sms/1/reports", headers=headers)
```

```
res = conn.getresponse()
data = res.read()

print(data.decode("utf-8"))
```

Java

```
HttpResponse<String> response =
Unirest.get("http://107.20.199.106/restapi/sms/1/reports")
  .header("authorization", "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==")
  .header("accept",  "application/json")
  .asString();
```

C#

```
var client = new RestClient("http://107.20.199.106/restapi/sms/1/reports");

var request = new RestRequest(Method.GET);
request.AddHeader("accept",  "application/json");
request.AddHeader("authorization", "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==");

IRestResponse response = client.Execute(request);
```

JavaScript

```
var data = null;

var xhr = new XMLHttpRequest();
xhr.withCredentials = true;

xhr.addEventListener("readystatechange", function () {
  if (this.readyState === this.DONE) {
    console.log(this.responseText);
  }
});

xhr.open("GET", "http://107.20.199.106/restapi/sms/1/reports");
xhr.setRequestHeader("authorization", "Basic
QWxhZGRpbjpvcGVuIHNlc2FtZQ==");
xhr.setRequestHeader("accept",  "application/json");

xhr.send(data);
```

Response

JSON

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

{
   "results":[
      {
         "bulkId":"8c20f086-d82b-48cc-b2b3-3ca5f7aca9fb",
         "messageId":"ff4804ef-6ab6-4abd-984d-ab3b1387e852",
         "to":"385981178",
         "sentAt":"2015-02-12T09:58:20.323+0100",
         "doneAt":"2015-02-12T09:58:20.337+0100",
         "smsCount":1,
         "price":{
            "pricePerMessage":0.01,
            "currency":"EUR"
         },
         "status":{
            "id":5,
            "groupId":3,
            "groupName":"DELIVERED",
            "name":"DELIVERED_TO_HANDSET",
            "description":"Message delivered to handset"
         },
         "error":{
            "groupId":0,
            "groupName":"OK",
            "id":0,
            "name":"NO_ERROR",
            "description":"No Error",
            "permanent":false
         }
      }
   ]
}
```

XML

```
HTTP/1.1 200 OK
```

```
Content-Type: application/xml

<SMSReportResponse>
   <results>
      <bulkId>8c20f086-d82b-48cc-b2b3-3ca5f7aca9fb</bulkId>
      <messageId>ff4804ef-6ab6-4abd-984d-ab3b1387e852</messageId>
      <to>385981178</to>
      <sentAt>2015-02-12T09:58:20.323+0100</sentAt>
      <doneAt>2015-02-12T09:58:20.337+0100</doneAt>
      <smsCount>1</smsCount>
      <price>
         <pricePerMessage>0.01</pricePerMessage>
         <currency>EUR</currency>
      </price>
      <status>
         <id>5</id>
         <groupId>3</groupId>
         <groupName>DELIVERED</groupName>
         <name>DELIVERED_TO_HANDSET</name>
         <description>Message delivered to handset</description>
      </status>
   </results>
</SMSReportResponse>
```

Getting the initial two delivery reports

Request

JSON

```
GET /restapi/sms/1/reports?limit=2 HTTP/1.1
Host: 107.20.199.106
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
Accept: application/json
```

XML

```
GET /restapi/sms/1/reports?limit=2 HTTP/1.1
Host: 107.20.199.106
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
Accept: application/xml
```

cURL

```
curl -X GET
-H 'Accept: application/json'
-H "Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ=="
http://107.20.199.106/restapi/sms/1/reports?limit=2
```

PHP

```
<?php

$request = new HttpRequest();
$request->setUrl('http://107.20.199.106/restapi/sms/1/reports');
$request->setMethod(HTTP_METH_GET);

$request->setQueryData(array(
  'limit' => '2'
));

$request->setHeaders(array(
  'accept' => 'application/json',
  'authorization' => 'Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ=='
));

try {
  $response = $request->send();

  echo $response->getBody();
} catch (HttpException $ex) {
  echo $ex;
}
```

Ruby

```
require 'uri'
require 'net/http'

url = URI("http://107.20.199.106/restapi/sms/1/reports?limit=2")

http = Net::HTTP.new(url.host, url.port)
http.use_ssl = true
http.verify_mode = OpenSSL::SSL::VERIFY_NONE
```

```
request =Net::HTTP::Get.new(url)
request["authorization"] = 'Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ=='
request["accept"] ='application/json'

response =http.request(request)
puts response.read_body
```

Python

```
import http.client

conn = http.client.httpConnection("107.20.199.106")

headers = {
    'authorization': "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==",
    'accept': "application/json"
    }

conn.request("GET", "/restapi/sms/1/reports?limit=2", headers=headers)

res = conn.getresponse()
data = res.read()

print(data.decode("utf-8"))
```

Java

```
HttpResponse<String> response =
Unirest.get("http://107.20.199.106/restapi/sms/1/reports?limit=2")
  .header("authorization", "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==")
  .header("accept",  "application/json")
  .asString();
```

C#

```
var client = new RestClient("http://107.20.199.106/restapi/sms/1/reports?
limit=2");

var request = new RestRequest(Method.GET);
request.AddHeader("accept",  "application/json");
request.AddHeader("authorization",  "Basic  QWxhZGRpbjpvcGVuIHNlc2FtZQ==");
```

```
IRestResponse response = client.Execute(request);
```

JavaScript

```
var data = null;

var xhr = new XMLHttpRequest();
xhr.withCredentials = true;

xhr.addEventListener("readystatechange", function () {
  if (this.readyState === this.DONE) {
    console.log(this.responseText);
  }
});

xhr.open("GET", "http://107.20.199.106/restapi/sms/1/reports?limit=2");
xhr.setRequestHeader("authorization", "Basic
QWxhZGRpbjpvcGVuIHNlc2FtZQ==");
xhr.setRequestHeader("accept", "application/json");

xhr.send(data);
```

Response

JSON

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

{
   "results":[
      {
         "bulkId":"80664c0c-e1ca-414d-806a-5caf146463df",
         "messageId":"bcfb828b-7df9-4e7b-8715-f34f5c61271a",
         "to":"38598111",
         "sentAt":"2015-02-12T09:58:20.323+0100",
         "doneAt":"2015-02-12T09:58:20.337+0100",
         "smsCount":1,
         "price":{
            "pricePerMessage":0.01,
            "currency":"EUR"
         },
```

```
            "status":{
               "groupId":3,
               "groupName":"DELIVERED",
               "id":5,
               "name":"DELIVERED_TO_HANDSET",
               "description":"Message delivered to handset"
            },
            "error":{
               "groupId":0,
               "groupName":"OK",
               "id":0,
               "name":"NO_ERROR",
               "description":"No Error",
               "permanent":false
            }
         },
         {
            "bulkId":"08fe4407-c48f-4d4b-a2f4-9ff583c985b8",
            "messageId":"12db39c3-7822-4e72-a3ec-c87442c0ffc5",
            "to":"385981112",
            "sentAt":"2015-02-12T09:58:20.345+0100",
            "doneAt":"2015-02-12T09:58:20.350+0100",
            "smsCount":1,
            "price":{
               "pricePerMessage":0.01,
               "currency":"EUR"
            },
            "status":{
               "groupId":3,
               "groupName":"DELIVERED",
               "id":5,
               "name":"DELIVERED_TO_HANDSET",
               "description":"Message delivered to handset"
            },
            "error":{
               "groupId":0,
               "groupName":"OK",
               "id":0,
               "name":"NO_ERROR",
               "description":"No Error",
               "permanent":false
            }
         }
      ]
   }
```

XML

```
HTTP/1.1 200 OK
Content-Type: application/xml

<SMSReportResponse>
   <results>
      <bulkId>80664c0c-e1ca-414d-806a-5caf146463df</bulkId>
      <messageId>bcfb828b-7df9-4e7b-8715-f34f5c61271a</messageId>
      <to>38598111</to>
      <sentAt>2015-02-12T09:58:20.323+0100</sentAt>
      <doneAt>2015-02-12T09:58:20.337+0100</doneAt>
      <smsCount>1</smsCount>
      <price>
         <pricePerMessage>0.01</pricePerMessage>
         <currency>EUR</currency>
      </price>
      <status>
         <groupId>3</groupId>
         <groupName>DELIVERED</groupName>
         <id>5</id>
         <name>DELIVERED_TO_HANDSET</name>
         <description>Message delivered to handset</description>
      </status>
      <error>
         <groupId>0</groupId>
         <groupName>OK</groupName>
         <id>0</id>
         <name>NO_ERROR</name>
         <description>No  Error</description>
         <permanent>false</permanent>
      </error>
   </results>
   <results>
      <bulkId>08fe4407-c48f-4d4b-a2f4-9ff583c985b8</bulkId>
      <messageId>12db39c3-7822-4e72-a3ec-c87442c0ffc5</messageId>
      <to>385981112</to>
      <sentAt>2015-02-12T09:58:20.345+0100</sentAt>
      <doneAt>2015-02-12T09:58:20.350+0100</doneAt>
      <smsCount>1</smsCount>
      <price>
         <pricePerMessage>0.01</pricePerMessage>
         <currency>EUR</currency>
      </price>
      <status>
```

```
            <groupId>3</groupId>
            <groupName>DELIVERED</groupName>
            <id>5</id>
            <name>DELIVERED_TO_HANDSET</name>
            <description>Message delivered to handset</description>
         </status>
         <error>
            <groupId>0</groupId>
            <groupName>OK</groupName>
            <id>0</id>
            <name>NO_ERROR</name>
            <description>No  Error</description>
            <permanent>false</permanent>
         </error>
      </results>
   </SMSReportResponse>
```

Sent messages logs

This method allows you to get logs for sent SMS.

Definition

```
http://107.20.199.106/restapi/sms/1/logs
```

Parameters

Parameter	Type	Description
from	String	Sender ID that can be alphanumeric or numeric.
to	String	The message destination address.
bulkId	String	The ID that uniquely identifies the request. Bulk ID will be received only when you send a message to more than one destination address.
messageId	String	The ID that uniquely identifies the message sent.
generalStatus	String	Sent SMS status group.Indicates whether the message is successfully sent, not sent, delivered, not delivered, waiting for delivery or any other possible status.

sentSince	DateTime	Lower limit on date and time of sending SMS.
sentUntil	DateTime	Upper limit on date and time of sending SMS.
limit	Integer	Maximal number of messages in returned logs.
mcc	String	Mobile country code.
mnc	String	Mobile network code.

Examples

```
GET /restapi/sms/1/logs HTTP/1.1
Host: 107.20.199.106
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
Accept: application/json
```

Result Format


```
{
   "results":[
      {
         "bulkId":"06479ba3-5977-47f6-9346-fee0369bc76b",
         "messageId":"1f21d8d7-f306-4f53-9f6e-eddfce9849ea",
         "to":"41793026727",
         "from":"InfoSMS",
         "text":"Test SMS.",
         "sentAt":"2015-02-23T17:40:31.773+0100",
         "doneAt":"2015-02-23T17:40:31.787+0100",
         "smsCount":1,
         "mccmnc":"22801",
         "price":{
            "pricePerMessage":0.01,
            "currency":"EUR"
         },
         "status":{
            "groupId":3,
            "groupName":"DELIVERED",
            "id":5,
            "name":"DELIVERED_TO_HANDSET",
            "description":"Message delivered to handset"
```

```
            },
            "error":{
               "groupId":0,
               "groupName":"OK",
               "id":0,
               "name":"NO_ERROR",
               "description":"No Error",
               "permanent":false
            }
         }
      ]
   }
```


Responseformat

> **Important:**
>
> **SMS logs are available for the last 48 hours!**

If successful, response header HTTP status code will be `200 OK` and the message logs will be returned.

If you try to send message without authorization, you will get a response with HTTP status code `401 Unauthorized`.

SMSLogsResponse

Parameter	Type	Description
results	SentSMSLog[]	Collection of logs.

SentSMSLog

Parameter	Type	Description
bulkId	String	The ID that uniquely identifies the request.
messageId	String	The ID that uniquely identifies the message sent.
to	String	The message destination address.
from	String	Sender ID that can be alphanumeric or numeric.

text	String	Text of the message that was sent.
sentAt	Date	Tells when the SMS was sent. Has the following format: `yyyy-MM-dd'T'HH:mm:ss.SSSZ` .
doneAt	Date	Tells when the SMS was finished processing by the platform (ie. delivered to destination, delivered to destination network, etc.)
smsCount	int	The number of sent message segments.
mccmnc	String	Mobile country and network codes.
price	Price	Sent SMS price.
status	Status	Indicates whether the message is successfully sent, not sent, delivered, not delivered, waiting for delivery or any other possible status.
error	Error	Indicates whether the error occurred during the query execution.

Price

Parameter	Type	Description
pricePerMessage	BigDecimal	Price per one SMS.
currency	String	The currency in which the price is expressed.

Status

Parameter	Type	Description
groupId	int	Status group ID.
groupName	String	Status group name.
id	int	Status ID.
name	String	Status name.
description	String	Human readable description of the status.
action	String	Action that should be taken to eliminate the error.

Error

Parameter	Type	Description
groupId	int	Error group ID.
groupName	String	Error group name.
id	int	Error ID.
name	String	Error name.
description	String	Human readable description of the error.
permanent	boolean	Tells if the error is permanent.

Additional examples

Getting logs without any query parameter

Request

JSON

```
GET /restapi/sms/1/logs HTTP/1.1
Host: 107.20.199.106
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
Accept: application/json
```

XML

```
GET /restapi/sms/1/logs HTTP/1.1
Host: 107.20.199.106
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
Accept: application/xml
```

cURL

```
curl -X GET
-H 'Accept: application/json'
-H 'Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ=='
http://107.20.199.106/restapi/sms/1/logs
```

PHP

```php
<?php

$request = new HttpRequest();
$request->setUrl('http://107.20.199.106/restapi/sms/1/logs');
$request->setMethod(HTTP_METH_GET);

$request->setHeaders(array(
  'accept' => 'application/json',
  'authorization' => 'Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ=='
));

try {
  $response = $request->send();

  echo $response->getBody();
} catch (HttpException $ex) {
  echo $ex;
}
```

Ruby

```ruby
require 'uri'
require 'net/http'

url = URI("http://107.20.199.106/restapi/sms/1/logs")

http = Net::HTTP.new(url.host, url.port)
http.use_ssl = true
http.verify_mode = OpenSSL::SSL::VERIFY_NONE

request = Net::HTTP::Get.new(url)
request["authorization"] = 'Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ=='
request["accept"] = 'application/json'

response = http.request(request)
puts response.read_body
```

Python

```
import http.client

conn = http.client.httpConnection("107.20.199.106")

headers = {
    'authorization': "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==",
    'accept': "application/json"
    }

conn.request("GET", "/restapi/sms/1/logs", headers=headers)

res = conn.getresponse()
data = res.read()

print(data.decode("utf-8"))
```

Java

```
HttpResponse<String> response =
Unirest.get("http://107.20.199.106/restapi/sms/1/logs")
  .header("authorization", "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==")
  .header("accept",  "application/json")
  .asString();
```

C#

```
var client = new RestClient("http://107.20.199.106/restapi/sms/1/logs");

var request = new RestRequest(Method.GET);
request.AddHeader("accept",  "application/json");
request.AddHeader("authorization", "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==");

IRestResponse response = client.Execute(request);
```

JavaScript

```
var data = null;

var xhr = new XMLHttpRequest();
xhr.withCredentials = true;
```

```
xhr.addEventListener("readystatechange", function () {
  if (this.readyState === this.DONE) {
    console.log(this.responseText);
  }
});

xhr.open("GET", "http://107.20.199.106/restapi/sms/1/logs");
xhr.setRequestHeader("authorization", "Basic
QWxhZGRpbjpvcGVuIHNlc2FtZQ==");
xhr.setRequestHeader("accept",  "application/json");

xhr.send(data);
```

Response

JSON

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

{
   "results":[
      {
         "bulkId":"bafdeb3d-719b-4cce-8762-54d47b40f3c5",
         "messageId":"07e03aae-fabc-44ad-b1ce-222e14094d70",
         "to":"41793026727",
         "from":"InfoSMS",
         "text":"Test SMS.",
         "sentAt":"2015-02-23T17:41:11.833+0100",
         "doneAt":"2015-02-23T17:41:11.843+0100",
         "smsCount":1,
         "mccmnc":"22801",
         "price":{
            "pricePerMessage":0.01,
            "currency":"EUR"
         },
         "status":{
            "groupId":3,
            "groupName":"DELIVERED",
            "id":5,
            "name":"DELIVERED_TO_HANDSET",
            "description":"Message delivered to handset"
         },
         "error":{
```

```
            "groupId":0,
            "groupName":"OK",
            "id":0,
            "name":"NO_ERROR",
            "description":"No Error",
            "permanent":false
         }
      },
      {
         "bulkId":"06479ba3-5977-47f6-9346-fee0369bc76b",
         "messageId":"1f21d8d7-f306-4f53-9f6e-eddfce9849ea",
         "to":"41793026727",
         "from":"InfoSMS",
         "text":"Test SMS.",
         "sentAt":"2015-02-23T17:40:31.773+0100",
         "doneAt":"2015-02-23T17:40:31.787+0100",
         "smsCount":1,
         "mccmnc":"22801",
         "price":{
            "pricePerMessage":0.01,
            "currency":"EUR"
         },
         "status":{
            "groupId":3,
            "groupName":"DELIVERED",
            "id":5,
            "name":"DELIVERED_TO_HANDSET",
            "description":"Message delivered to handset"
         },
         "error":{
            "groupId":0,
            "groupName":"OK",
            "id":0,
            "name":"NO_ERROR",
            "description":"No Error",
            "permanent":false
         }
      }
   ]
}
```


XML

```
HTTP/1.1 200 OK
```

```
Content-Type: application/xml

<SMSLogsResponse>
    <results>
        <results>
            <bulkId>7944c32d-bf77-4f41-a752-c3aa89027adc</bulkId>
            <messageId>f97d3b99-fab2-468e-8acf-c8c8792b8ce6</messageId>
            <to>41793026727</to>
            <from>InfoSMS</from>
            <text>Test SMS.</text>
            <sentAt>2015-02-23T17:41:18.020+0100</sentAt>
            <doneAt>2015-02-23T17:41:18.027+0100</doneAt>
            <smsCount>1</smsCount>
            <mccmnc>22801</mccmnc>
            <price>
                <pricePerMessage>0.0100</pricePerMessage>
                <currency>EUR</currency>
            </price>
            <status>
                <groupId>3</groupId>
                <groupName>DELIVERED</groupName>
                <id>5</id>
                <name>DELIVERED_TO_HANDSET</name>
                <description>Message delivered to handset</description>
            </status>
            <error>
                <groupId>0</groupId>
                <groupName>Ok</groupName>
                <id>0</id>
                <name>NO_ERROR</name>
                <description>No  Error</description>
                <permanent>false</permanent>
            </error>
        </results>
        <results>
            <bulkId>bafdeb3d-719b-4cce-8762-54d47b40f3c5</bulkId>
            <messageId>07e03aae-fabc-44ad-b1ce-222e14094d70</messageId>
            <to>41793026727</to>
            <from>InfoSMS</from>
            <text>Test SMS.</text>
            <sentAt>2015-02-23T17:41:11.833+0100</sentAt>
            <doneAt>2015-02-23T17:41:11.843+0100</doneAt>
            <smsCount>1</smsCount>
            <mccmnc>22801</mccmnc>
            <price>
                <pricePerMessage>0.0100</pricePerMessage>
```

```
                <currency>EUR</currency>
            </price>
            <status>
                <groupId>3</groupId>
                <groupName>DELIVERED</groupName>
                <id>5</id>
                <name>DELIVERED_TO_HANDSET</name>
                <description>Message delivered to handset</description>
            </status>
            <error>
                <groupId>0</groupId>
                <groupName>Ok</groupName>
                <id>0</id>
                <name>NO_ERROR</name>
                <description>No  Error</description>
                <permanent>false</permanent>
            </error>
        </results>
    </results>
</SMSLogsResponse>
```

Getting logs with `from`, `to` and `limit` as filters

Request

JSON

```
GET  /restapi/sms/1/logs?from=InfoSMS&to=41793026727&limit=1  HTTP/1.1
Host: 107.20.199.106
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
Accept: application/json
```

XML

```
POST  /restapi/sms/1/logs?from=InfoSMS&to=41793026727&limit=1  HTTP/1.1
Host: 107.20.199.106
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
Accept: application/xml
```

cURL

```
curl -X GET \
-H 'Accept: application/json' \
-H 'Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==' \
http://107.20.199.106/restapi/sms/1/logs?
from=InfoSMS&to=41793026727&limit=1
```

PHP

```
<?php

$request = new HttpRequest();
$request->setUrl('http://107.20.199.106/restapi/sms/1/logs');
$request->setMethod(HTTP_METH_GET);

$request->setQueryData(array(
  'from' => 'InfoSMS',
  'to' => '41793026727',
  'limit' => '1'
));

$request->setHeaders(array(
  'accept' => 'application/json',
  'authorization' => 'Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ=='
));

try {
  $response = $request->send();

  echo $response->getBody();
} catch (HttpException $ex) {
  echo $ex;
}
```

Ruby

```
require 'uri'
require 'net/http'

url = URI("http://107.20.199.106/restapi/sms/1/logs?
from=InfoSMS&to=41793026727&limit=1")

http = Net::HTTP.new(url.host, url.port)
```

```
http.use_ssl = true
http.verify_mode = OpenSSL::SSL::VERIFY_NONE

request = Net::HTTP::Get.new(url)
request["authorization"] = 'Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ=='
request["accept"] = 'application/json'

response = http.request(request)
puts response.read_body
```

Python

```
import http.client

conn = http.client.httpConnection("107.20.199.106")

headers = {
    'authorization': "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==",
    'accept': "application/json"
    }

conn.request("GET", "/restapi/sms/1/logs?
from=InfoSMS&to=41793026727&limit=1",   headers=headers)

res = conn.getresponse()
data = res.read()

print(data.decode("utf-8"))
```

Java

```
HttpResponse<String> response =
Unirest.get("http://107.20.199.106/restapi/sms/1/logs?
from=InfoSMS&to=41793026727&limit=1")
  .header("authorization", "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==")
  .header("accept",  "application/json")
  .asString();
```

C#

```
var client = new RestClient("http://107.20.199.106/restapi/sms/1/logs?
from=InfoSMS&to=41793026727&limit=1");
```

```
var request = new RestRequest(Method.GET);
request.AddHeader("accept",  "application/json");
request.AddHeader("authorization", "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==");

IRestResponse response = client.Execute(request);
```

JavaScript

```
var data = null;

var xhr = new XMLHttpRequest();
xhr.withCredentials = true;

xhr.addEventListener("readystatechange", function () {
  if (this.readyState === this.DONE) {
    console.log(this.responseText);
  }
});

xhr.open("GET", "http://107.20.199.106/restapi/sms/1/logs?
from=InfoSMS&to=41793026727&limit=1");
xhr.setRequestHeader("authorization", "Basic
QWxhZGRpbjpvcGVuIHNlc2FtZQ==");
xhr.setRequestHeader("accept",  "application/json");

xhr.send(data);
```

Response

JSON

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

{
   "results":[
      {
         "bulkId":"82d1d36e-e4fb-4194-8b93-caeb053bd327",
         "messageId":"fc0cbfb8-7a72-40da-a76d-e2c2d9400835",
         "to":"41793026727",
         "from":"InfoSMS",
         "text":"Test SMS.",
```

```
            "sentAt":"2015-02-23T17:42:05.390+0100",
            "doneAt":"2015-02-23T17:42:05.390+0100",
            "smsCount":1,
            "mccmnc":"22801",
            "price":{
               "pricePerMessage":0,
               "currency":"EUR"
            },
            "status":{
               "groupId":5,
               "groupName":"REJECTED",
               "id":6,
               "name":"REJECTED_NETWORK",
               "description":"Network is forbidden",
               "action":"Contact account manager"
            },
            "error":{
               "groupId":0,
               "groupName":"OK",
               "id":0,
               "name":"NO_ERROR",
               "description":"No Error",
               "permanent":false
            }
         }
      ]
   }
```


XML


```
   HTTP/1.1 200 OK
   Content-Type: application/xml

   <SMSLogsResponse>
      <results>
         <results>
            <bulkId>82d1d36e-e4fb-4194-8b93-caeb053bd327</bulkId>
            <messageId>fc0cbfb8-7a72-40da-a76d-e2c2d9400835</messageId>
            <to>41793026727</to>
            <from>InfoSMS</from>
            <text>Test SMS.</text>
            <sentAt>2015-02-23T17:42:05.390+0100</sentAt>
            <doneAt>2015-02-23T17:42:05.390+0100</doneAt>
            <smsCount>1</smsCount>
```

```
            <mccmnc>22801</mccmnc>
            <price>
               <pricePerMessage>0.0000</pricePerMessage>
               <currency>EUR</currency>
            </price>
            <status>
               <groupId>5</groupId>
               <groupName>REJECTED</groupName>
               <id>6</id>
               <name>REJECTED_NETWORK</name>
               <description>Network is forbidden</description>
               <action>Contact account manager</action>
            </status>
            <error>
               <groupId>0</groupId>
               <groupName>OK</groupName>
               <id>0</id>
               <name>NO_ERROR</name>
               <description>No  Error</description>
               <permanent>false</permanent>
            </error>
         </results>
      </results>
   </SMSLogsResponse>
```

Getting logs with multiple `bulkIds` as filters

Request

JSON

```
GET /restapi/sms/1/logs?bulkId=1dece649-6c8f-404e-8c6e-
c7e073be509a,bafdeb3d-719b-4cce-8762-54d47b40f3c5 HTTP/1.1
Host: 107.20.199.106
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
Accept: application/json
```

XML

```
POST /restapi/sms/1/logs?bulkId=1dece649-6c8f-404e-8c6e-
c7e073be509a,bafdeb3d-719b-4cce-8762-54d47b40f3c5 HTTP/1.1
Host: 107.20.199.106
```

```
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
Accept: application/xml
```

cURL

```
curl -X GET
-H 'Accept: application/json'
-H 'Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ=='
http://107.20.199.106/restapi/sms/1/logs?bulkId=1dece649-6c8f-404e-8c6e-
c7e073be509a,bafdeb3d-719b-4cce-8762-54d47b40f3c5
```

PHP

```
<?php

$request = new HttpRequest();
$request->setUrl('http://107.20.199.106/restapi/sms/1/logs');
$request->setMethod(HTTP_METH_GET);

$request->setQueryData(array(
  'bulkId' => '1dece649-6c8f-404e-8c6e-c7e073be509a,bafdeb3d-719b-4cce-
8762-54d47b40f3c5'
));

$request->setHeaders(array(
  'accept' => 'application/json',
  'authorization' => 'Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ=='
));

try {
  $response = $request->send();

  echo $response->getBody();
} catch (HttpException $ex) {
  echo $ex;
}
```

Ruby

```
require 'uri'
require 'net/http'
```

```
url = URI("http://107.20.199.106/restapi/sms/1/logs?bulkId=1dece649-6c8f-
404e-8c6e-c7e073be509a%2Cbafdeb3d-719b-4cce-8762-54d47b40f3c5")

http = Net::HTTP.new(url.host, url.port)
http.use_ssl = true
http.verify_mode = OpenSSL::SSL::VERIFY_NONE

request =Net::HTTP::Get.new(url)
request["authorization"] = 'Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ=='
request["accept"] ='application/json'

response =http.request(request)
puts response.read_body
```

Python

```
import http.client

conn = http.client.httpConnection("107.20.199.106")

headers = {
    'authorization': "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==",
    'accept': "application/json"
    }

conn.request("GET", "/restapi/sms/1/logs?bulkId=1dece649-6c8f-404e-8c6e-
c7e073be509a%2Cbafdeb3d-719b-4cce-8762-54d47b40f3c5",   headers=headers)

res = conn.getresponse()
data = res.read()

print(data.decode("utf-8"))
```

Java

```
HttpResponse<String> response =
Unirest.get("http://107.20.199.106/restapi/sms/1/logs?bulkId=1dece649-6c8f-
404e-8c6e-c7e073be509a%2Cbafdeb3d-719b-4cce-8762-54d47b40f3c5")
  .header("authorization", "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==")
  .header("accept",  "application/json")
  .asString();
```

C#

```
var client = new RestClient("http://107.20.199.106/restapi/sms/1/logs?
bulkId=1dece649-6c8f-404e-8c6e-c7e073be509a%2Cbafdeb3d-719b-4cce-8762-
54d47b40f3c5");

var request = new RestRequest(Method.GET);
request.AddHeader("accept",  "application/json");
request.AddHeader("authorization", "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==");

IRestResponse response = client.Execute(request);
```

JavaScript

```
var data = null;

var xhr = new XMLHttpRequest();
xhr.withCredentials = true;

xhr.addEventListener("readystatechange", function () {
  if (this.readyState === this.DONE) {
    console.log(this.responseText);
  }
});

xhr.open("GET", "http://107.20.199.106/restapi/sms/1/logs?bulkId=1dece649-
6c8f-404e-8c6e-c7e073be509a%2Cbafdeb3d-719b-4cce-8762-54d47b40f3c5");
xhr.setRequestHeader("authorization", "Basic
QWxhZGRpbjpvcGVuIHNlc2FtZQ==");
xhr.setRequestHeader("accept", "application/json");

xhr.send(data);
```

Response

JSON

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

{
   "results":[
```

```
{
   "bulkId":"bafdeb3d-719b-4cce-8762-54d47b40f3c5",
   "messageId":"07e03aae-fabc-44ad-b1ce-222e14094d70",
   "to":"41793026727",
   "from":"InfoSMS",
   "text":"Test SMS.",
   "sentAt":"2015-02-23T17:41:11.833+0100",
   "doneAt":"2015-02-23T17:41:11.843+0100",
   "smsCount":1,
   "mccmnc":"22801",
   "price":{
      "pricePerMessage":0.01,
      "currency":"EUR"
   },
   "status":{
      "groupId":3,
      "groupName":"DELIVERED",
      "id":5,
      "name":"DELIVERED_TO_HANDSET",
      "description":"Message delivered to handset"
   },
   "error":{
      "groupId":0,
      "groupName":"OK",
      "id":0,
      "name":"NO_ERROR",
      "description":"No Error",
      "permanent":false
   }
},
{
   "bulkId":"1dece649-6c8f-404e-8c6e-c7e073be509a",
   "messageId":"faa48fe6-fe2c-4f36-a43b-a070e2906ecb",
   "to":"41793026727",
   "from":"InfoSMS",
   "text":"Test SMS.",
   "sentAt":"2015-02-23T16:22:37.413+0100",
   "doneAt":"2015-02-23T16:22:37.437+0100",
   "smsCount":1,
   "mccmnc":"22801",
   "price":{
      "pricePerMessage":0,
      "currency":"EUR"
   },
   "status":{
      "groupId":2,
```

```
            "groupName":"UNDELIVERABLE",
            "id":9,
            "name":"UNDELIVERABLE_NOT_DELIVERED",
            "description":"Message sent not delivered"
         },
         "error":{
            "groupId":0,
            "groupName":"OK",
            "id":0,
            "name":"NO_ERROR",
            "description":"No Error",
            "permanent":false
         }
      }
   ]
}
```


XML


```
HTTP/1.1 200 OK
Content-Type: application/xml

<SMSLogsResponse>
   <results>
      <results>
         <bulkId>bafdeb3d-719b-4cce-8762-54d47b40f3c5</bulkId>
         <messageId>07e03aae-fabc-44ad-b1ce-222e14094d70</messageId>
         <to>41793026727</to>
         <from>InfoSMS</from>
         <text>Test SMS.</text>
         <sentAt>2015-02-23T17:41:11.833+0100</sentAt>
         <doneAt>2015-02-23T17:41:11.843+0100</doneAt>
         <smsCount>1</smsCount>
         <mccmnc>22801</mccmnc>
         <price>
            <pricePerMessage>0.0100</pricePerMessage>
            <currency>EUR</currency>
         </price>
         <status>
            <groupId>3</groupId>
            <groupName>DELIVERED</groupName>
            <id>5</id>
            <name>DELIVERED_TO_HANDSET</name>
            <description>Message delivered to handset</description>
```

```
            </status>
            <error>
               <groupId>0</groupId>
               <groupName>OK</groupName>
               <id>0</id>
               <name>NO_ERROR</name>
               <description>No  Error</description>
               <permanent>false</permanent>
            </error>
         </results>
         <results>
            <bulkId>1dece649-6c8f-404e-8c6e-c7e073be509a</bulkId>
            <messageId>faa48fe6-fe2c-4f36-a43b-a070e2906ecb</messageId>
            <to>41793026727</to>
            <from>InfoSMS</from>
            <text>Test SMS.</text>
            <sentAt>2015-02-23T16:22:37.413+0100</sentAt>
            <doneAt>2015-02-23T16:22:37.437+0100</doneAt>
            <smsCount>1</smsCount>
            <mccmnc>22801</mccmnc>
            <price>
               <pricePerMessage>0.0000</pricePerMessage>
               <currency>EUR</currency>
            </price>
            <status>
               <groupId>2</groupId>
               <groupName>UNDELIVERABLE</groupName>
               <id>9</id>
               <name>UNDELIVERABLE_NOT_DELIVERED</name>
               <description>Message sent not delivered</description>
            </status>
            <error>
               <groupId>0</groupId>
               <groupName>OK</groupName>
               <id>0</id>
               <name>NO_ERROR</name>
               <description>No  Error</description>
               <permanent>false</permanent>
            </error>
         </results>
      </results>
   </SMSLogsResponse>
```


Getting logs with `date range` and `general status` as filters

Request

JSON

```
GET /restapi/sms/1/logs?sentSince=2015-02-
22T17:42:05.390%2b01:00&generalStatus=DELIVERED HTTP/1.1
Host: 107.20.199.106
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
Accept: application/json
```

XML

```
POST /restapi/sms/1/logs?sentSince=2015-02-
22T17:42:05.390%2b01:00&generalStatus=DELIVERED HTTP/1.1
Host: 107.20.199.106
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
Accept: application/xml
```

cURL

```
curl -X GET
-H 'Accept: application/json'
-H 'Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ=='
http://107.20.199.106/restapi/sms/1/logs?sentSince=2015-02-
22T17:42:05.390%2b01:00&generalStatus=DELIVERED
```

PHP

```
<?php

$request = new HttpRequest();
$request->setUrl('http://107.20.199.106/restapi/sms/1/logs');
$request->setMethod(HTTP_METH_GET);

$request->setQueryData(array(
  'sentSince' => '2015-02-22T17:42:05.390+01:00',
  'generalStatus' => 'DELIVERED'
));

$request->setHeaders(array(
  'accept' => 'application/json',
```

```
    'authorization' => 'Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ=='
));

try {
  $response = $request->send();

  echo $response->getBody();
} catch (HttpException $ex) {
  echo $ex;
}
```

Ruby

```
require 'uri'
require 'net/http'

url = URI("http://107.20.199.106/restapi/sms/1/logs?sentSince=2015-02-
22T17%3A42%3A05.390%2B01%3A00&generalStatus=DELIVERED")

http = Net::HTTP.new(url.host, url.port)
http.use_ssl = true
http.verify_mode = OpenSSL::SSL::VERIFY_NONE

request = Net::HTTP::Get.new(url)
request["authorization"] = 'Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ=='
request["accept"] = 'application/json'

response = http.request(request)
puts response.read_body
```

Python

```
import http.client

conn = http.client.httpConnection("107.20.199.106")

headers = {
    'authorization': "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==",
    'accept': "application/json"
    }

conn.request("GET", "/restapi/sms/1/logs?sentSince=2015-02-
22T17%3A42%3A05.390%2B01%3A00&generalStatus=DELIVERED",   headers=headers)
```

```
res = conn.getresponse()
data = res.read()

print(data.decode("utf-8"))
```

Java

```
HttpResponse<String> response =
Unirest.get("http://107.20.199.106/restapi/sms/1/logs?sentSince=2015-02-
22T17%3A42%3A05.390%2B01%3A00&generalStatus=DELIVERED")
  .header("authorization", "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==")
  .header("accept",  "application/json")
  .asString();
```

C#

```
var client = new RestClient("http://107.20.199.106/restapi/sms/1/logs?
sentSince=2015-02-22T17%3A42%3A05.390%2B01%3A00&generalStatus=DELIVERED");

var request = new RestRequest(Method.GET);
request.AddHeader("accept",  "application/json");
request.AddHeader("authorization", "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==");

IRestResponse response = client.Execute(request);
```

JavaScript

```
var data = null;

var xhr = new XMLHttpRequest();
xhr.withCredentials = true;

xhr.addEventListener("readystatechange", function () {
  if (this.readyState === this.DONE) {
    console.log(this.responseText);
  }
});

xhr.open("GET", "http://107.20.199.106/restapi/sms/1/logs?sentSince=2015-
02-22T17%3A42%3A05.390%2B01%3A00&generalStatus=DELIVERED");
xhr.setRequestHeader("authorization",  "Basic
```

```
QWxhZGRpbjpvcGVuIHNlc2FtZQ==");
xhr.setRequestHeader("accept",  "application/json");

xhr.send(data);
```

Response

JSON

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

{
   "results":[
      {
         "bulkId":"ce166d0e-bac7-4639-a094-52c406852afd",
         "messageId":"fdf71ada-3308-4cd1-9962-31f2b937a1d0",
         "to":"41793026727",
         "from":"InfoSMS",
         "text":"Test SMS.",
         "sentAt":"2015-02-24T10:34:56.463+0100",
         "doneAt":"2015-02-24T10:34:56.480+0100",
         "smsCount":1,
         "mccmnc":"22801",
         "price":{
            "pricePerMessage":0.01,
            "currency":"EUR"
         },
         "status":{
            "groupId":3,
            "groupName":"DELIVERED",
            "id":5,
            "name":"DELIVERED_TO_HANDSET",
            "description":"Message delivered to handset"
         },
         "error":{
            "groupId":0,
            "groupName":"OK",
            "id":0,
            "name":"NO_ERROR",
            "description":"No Error",
            "permanent":false
         }
      }
```

```
    ]
}
```

XML

```
HTTP/1.1 200 OK
Content-Type: application/xml

<SMSLogsResponse>
   <results>
      <results>
         <bulkId>ce166d0e-bac7-4639-a094-52c406852afd</bulkId>
         <messageId>fdf71ada-3308-4cd1-9962-31f2b937a1d0</messageId>
         <to>41793026727</to>
         <from>InfoSMS</from>
         <text>Test SMS.</text>
         <sentAt>2015-02-24T10:34:56.463+0100</sentAt>
         <doneAt>2015-02-24T10:34:56.480+0100</doneAt>
         <smsCount>1</smsCount>
         <mccmnc>22801</mccmnc>
         <price>
            <pricePerMessage>0.0100</pricePerMessage>
            <currency>EUR</currency>
         </price>
         <status>
            <groupId>3</groupId>
            <groupName>DELIVERED</groupName>
            <id>5</id>
            <name>DELIVERED_TO_HANDSET</name>
            <description>Message delivered to handset</description>
         </status>
         <error>
            <groupId>0</groupId>
            <groupName>OK</groupName>
            <id>0</id>
            <name>NO_ERROR</name>
            <description>No  Error</description>
            <permanent>false</permanent>
         </error>
      </results>
   </results>
</SMSLogsResponse>
```

Advanced SMS methods

Fully featured textual message

Send advanced SMS with the all available features and parameters.

Definition

```
http://107.20.199.106/restapi/sms/1/text/advanced
```

Parameters

Parameter	Type	Description
from	String	Represents a sender ID which can be alphanumeric or numeric. *Alphanumeric* sender ID length should be between 3 and 11 characters (Example: `CompanyName`). *Numeric* sender ID length should be between 3 and 14 characters.
to	String[]	**Required.** Array of message destination addresses. If you want to send a message to one destination, a single String is supported instead of an Array. Destination addresses must be in international format (Example: `41793026727`).
text	String	Text of the message that will be sent.
bulkId	String	The ID which uniquely identifies the request. Bulk ID will be received only when you send a message to more than one destination address.
messageId	String	The ID that uniquely identifies the message sent.
flash	Boolean	Can be `true` or `false` . If the value is set to `true` , a flash SMS will be sent. Otherwise, a normal SMS will be sent. The default value is `false` .
transliteration	String	Conversion of a message text from one script to another. Possible values: `"TURKISH"` , `"GREEK"` , `"CYRILLIC"` , `"CENTRAL_EUROPEAN"` .
languageCode	String	Code for language character set of a message text. Possible values: `TR` for Turkish, `ES` for Spanish and `PT` for Portuguese.

singleShift	Boolean	Single shift table replacing the GSM 7 bit default alphabet extension table. Find more details here.
lockingShift	Boolean	Locking shift table replacing standard GSM 7 bit default alphabet table. Find more details here.
notifyUrl	String	The URL on your call back server on which the Delivery report will be sent.
notifyContentType	String	Preferred Delivery report content type. Can be `application/json` or `application/xml`.
callbackData	String	Additional client's data that will be sent on the notifyUrl.
validityPeriod	Integer	The message validity period in minutes. When the period expires, it will not be allowed for the message to be sent. Validity period longer than 48h is not supported (in this case, it will be automatically set to 48h).
sendAt	DateTime	Date and time when the message is to be sent. Used for scheduled SMS (SMS not sent immediately, but at scheduled time).

Examples

JSON

```
POST /restapi/sms/1/text/advanced HTTP/1.1
Host: 107.20.199.106
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
Content-Type: application/json

{
   "bulkId":"BULK-ID-123-xyz",
   "messages":[
      {
         "from":"InfoSMS",
         "destinations":[
            {
               "to":"41793026727",
               "messageId":"MESSAGE-ID-123-xyz"
            },
            {
               "to":"41793026731"
```

```
            }
         ],
         "text":"Artık Ulusal Dil Tanımlayıcısı ile Türkçe karakterli
smslerinizi rahatlıkla iletebilirsiniz.",
         "flash":false,
         "language":{
            "languageCode":"TR",
            "singleShift":true,
            "lockingShift":false
         },
         "transliteration":"TURKISH",
         "notifyUrl":"http://www.example.com/sms/advanced",
         "notifyContentType":"application/json",
         "callbackData":"DLR callback data",
         "validityPeriod": 720
      },
      {
         "from":"41793026700",
         "destinations":[
            {
               "to":"41793026785"
            }
         ],
         "text":"A long time ago, in a galaxy far, far away... It is a
period of civil war. Rebel spaceships, striking from a hidden base, have
won their first victory against the evil Galactic Empire.",
         "sendAt":"2015-07-07T17:00:00.000+01:00"
      }
   ]
}
```

XML

```
POST /restapi/sms/1/text/advanced HTTP/1.1
Host: 107.20.199.106
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
Content-Type: application/xml

<request>
  <bulkId>BULK-ID-123-xyz</bulkId>
  <messages>
    <from>InfoSMS</from>
    <destinations>
      <to>41793026727</to>
```

```
      <messageId>MESSAGE-ID-123-xyz</messageId>
    </destinations>
    <destinations>
      <to>41793026731</to>
    </destinations>
    <text>Artık Ulusal Dil Tanımlayıcısı ile Türkçe karakterli smslerinizi
rahatlıkla  iletebilirsiniz.</text>
    <flash>false</flash>
    <language>
      <languageCode>TR</languageCode>
      <singleShift>true</singleShift>
      <lockingShift>false</lockingShift>
    </language>
    <transliteration>TURKISH</transliteration>
    <notifyUrl>http://www.example.com/sms/advanced</notifyUrl>
    <notifyContentType>application/json</notifyContentType>
    <callbackData>DLR callback data</callbackData>
    <validityPeriod>720</validityPeriod>
  </messages>
  <messages>
    <from>41793026700</from>
    <destinations>
      <to>41793026785</to>
    </destinations>
    <text>A long time ago, in a galaxy far, far away... It is a period of
civil war. Rebel spaceships, striking from a hidden base, have won their
first victory against the evil Galactic Empire.</text>
    <sendAt>2015-07-07T17:00:00.000+01:00</sendAt>
  </messages>
</request>
```

cURL

```
curl -X POST \
 -H 'Content-Type: application/json' \
 -H 'Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==' \
 -d '{
   "bulkId":"BULK-ID-123-xyz",
   "messages":[
      {
         "from":"InfoSMS",
         "destinations":[
            {
               "to":"41793026727",
```

```
                "messageId":"MESSAGE-ID-123-xyz"
            },
            {
                "to":"41793026731"
            }
        ],
        "text":"Artık Ulusal Dil Tanımlayıcısı ile Türkçe karakterli
smslerinizi rahatlıkla iletebilirsiniz.",
        "flash":false,
        "language":{
            "singleShift":true,
            "lockingShift":false,
            "languageCode":"TR"
        },
        "transliteration": "TURKISH"
        "notify":true,
        "notifyUrl":"http://www.example.com/sms/advanced",
        "notifyContentType":"application/json",
        "callbackData":"DLR callback data",
        "validityPeriod": 720
    },
    {
        "from":"41793026700",
        "destinations":[
            {
                "to":"41793026785"
            }
        ],
        "text":"A long time ago, in a galaxy far, far away... It is a
period of civil war. Rebel spaceships, striking from a hidden base, have
won their first victory against the evil Galactic Empire."
    },
        "sendAt":"2015-07-07T17:00:00.000+01:00"
  ]
}'  http://107.20.199.106/restapi/sms/1/text/advanced
```

PHP

```
<?php

$request = new HttpRequest();
$request->setUrl('http://107.20.199.106/restapi/sms/1/text/advanced');
$request->setMethod(HTTP_METH_POST);
```

```
$request->setHeaders(array(
  'accept' => 'application/json',
  'authorization' => 'Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==',
  'content-type' => 'application/json'
));

$request->setBody('{
   "bulkId":"BULK-ID-123-xyz",
   "messages":[
      {
         "from":"InfoSMS",
         "destinations":[
            {
               "to":"41793026727",
               "messageId":"MESSAGE-ID-123-xyz"
            },
            {
               "to":"41793026731"
            }
         ],
         "text":"Artık Ulusal Dil Tanımlayıcısı ile Türkçe karakterli
smslerinizi rahatlıkla iletebilirsiniz.",
         "flash":false,
         "language":{
            "languageCode":"TR",
            "singleShift":true,
            "lockingShift":false
         },
         "transliteration":"TURKISH",
         "notifyUrl":"http://www.example.com/sms/advanced",
         "notifyContentType":"application/json",
         "callbackData":"DLR callback data",
         "validityPeriod": 720
      },
      {
         "from":"41793026700",
         "destinations":[
            {
               "to":"41793026785"
            }
         ],
         "text":"A long time ago, in a galaxy far, far away... It is a
period of civil war. Rebel spaceships, striking from a hidden base, have
won their first victory against the evil Galactic Empire.",
         "sendAt":"2015-07-07T17:00:00.000+01:00"
      }
```

```
    ]
}');

try {
  $response = $request->send();

  echo $response->getBody();
} catch (HttpException $ex) {
  echo $ex;
}
```

Ruby

```
require 'uri'
require 'net/http'

url = URI("http://107.20.199.106/restapi/sms/1/text/advanced")

http = Net::HTTP.new(url.host, url.port)
http.use_ssl = true
http.verify_mode = OpenSSL::SSL::VERIFY_NONE

request = Net::HTTP::Post.new(url)
request["content-type"] = 'application/json'
request["authorization"] = 'Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ=='
request["accept"] ='application/json'

request.body = "{\"bulkId\":\"BULK-ID-123-xyz\", \"messages\":
[{\"from\":\"InfoSMS\",   \"destinations\":[{\"to\":\"41793026727\",
\"messageId\":\"MESSAGE-ID-123-xyz\"},   {\"to\":\"41793026731\"}],
\"text\":\"Artık Ulusal Dil Tanımlayıcısı ile Türkçe karakterli smslerinizi
rahatlıkla iletebilirsiniz.\", \"flash\":false, \"language\":
{\"languageCode\":\"TR\",  \"singleShift\":true,  \"lockingShift\":false},
\"transliteration\":\"TURKISH\",
\"notifyUrl\":\"http://www.example.com/sms/advanced\",
\"notifyContentType\":\"application/json\", \"callbackData\":\"DLR callback
data\", \"validityPeriod\": 720}, {\"from\":\"41793026700\",
\"destinations\":[{\"to\":\"41793026785\"}], \"text\":\"A long time ago, in
a galaxy far, far away... It is a period of civil war. Rebel spaceships,
striking from a hidden base, have won their first victory against the evil
Galactic Empire.\",\n  \t\t\t\"sendAt\":\"2015-07-
07T17:00:00.000+01:00\"}]}"

response =http.request(request)
```

```
puts response.read_body
```

Python

```
import http.client

conn = http.client.HTTPConnection("107.20.199.106")

payload = "{\"bulkId\":\"BULK-ID-123-xyz\", \"messages\":
[{\"from\":\"InfoSMS\",   \"destinations\":[{\"to\":\"41793026727\",
\"messageId\":\"MESSAGE-ID-123-xyz\"},   {\"to\":\"41793026731\"}],
\"text\":\"Artık Ulusal Dil Tanımlayıcısı ile Türkçe karakterli smslerinizi
rahatlıkla iletebilirsiniz.\", \"flash\":false, \"language\":
{\"languageCode\":\"TR\",  \"singleShift\":true,  \"lockingShift\":false},
\"transliteration\":\"TURKISH\",
\"notifyUrl\":\"http://www.example.com/sms/advanced\",
\"notifyContentType\":\"application/json\", \"callbackData\":\"DLR callback
data\", \"validityPeriod\": 720}, {\"from\":\"41793026700\",
\"destinations\":[{\"to\":\"41793026785\"}], \"text\":\"A long time ago, in
a galaxy far, far away... It is a period of civil war. Rebel spaceships,
striking from a hidden base, have won their first victory against the evil
Galactic Empire.\",\n  \t\t\t\"sendAt\":\"2015-07-
07T17:00:00.000+01:00\"}]}"

headers = {
    'content-type':  "application/json",
    'authorization': "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==",
    'accept': "application/json"
    }

conn.request("POST", "/restapi/sms/1/text/advanced", payload, headers)

res = conn.getresponse()
data = res.read()

print(data.decode("utf-8"))
```

Java

```
HttpResponse<String> response =
Unirest.post("http://107.20.199.106/restapi/sms/1/text/advanced")
  .header("content-type",  "application/json")
  .header("authorization", "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==")
```

```
  .header("accept",  "application/json")
  .body("{\"bulkId\":\"BULK-ID-123-xyz\", \"messages\":
[{\"from\":\"InfoSMS\",   \"destinations\":[{\"to\":\"41793026727\",
\"messageId\":\"MESSAGE-ID-123-xyz\"},   {\"to\":\"41793026731\"}],
\"text\":\"Artık Ulusal Dil Tanımlayıcısı ile Türkçe karakterli smslerinizi
rahatlıkla iletebilirsiniz.\", \"flash\":false, \"language\":
{\"languageCode\":\"TR\",  \"singleShift\":true,  \"lockingShift\":false},
\"transliteration\":\"TURKISH\",
\"notifyUrl\":\"http://www.example.com/sms/advanced\",
\"notifyContentType\":\"application/json\", \"callbackData\":\"DLR callback
data\", \"validityPeriod\": 720}, {\"from\":\"41793026700\",
\"destinations\":[{\"to\":\"41793026785\"}], \"text\":\"A long time ago, in
a galaxy far, far away... It is a period of civil war. Rebel spaceships,
striking from a hidden base, have won their first victory against the evil
Galactic Empire.\",\n  \t\t\t\"sendAt\":\"2015-07-
07T17:00:00.000+01:00\"}]}")
  .asString();
```

C#

```
var client = new
RestClient("http://107.20.199.106/restapi/sms/1/text/advanced");
var request = new RestRequest(Method.POST);
request.AddHeader("accept",  "application/json");
request.AddHeader("authorization", "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==");
request.AddHeader("content-type", "application/json");
request.AddParameter("application/json",   "{\"bulkId\":\"BULK-ID-123-xyz\",
\"messages\":[{\"from\":\"InfoSMS\", \"destinations\":
[{\"to\":\"41793026727\",   \"messageId\":\"MESSAGE-ID-123-xyz\"},
{\"to\":\"41793026731\"}], \"text\":\"Artık Ulusal Dil Tanımlayıcısı ile
Türkçe karakterli smslerinizi rahatlıkla iletebilirsiniz.\",
\"flash\":false,  \"language\":{\"languageCode\":\"TR\",
\"singleShift\":true,  \"lockingShift\":false},
\"transliteration\":\"TURKISH\",
\"notifyUrl\":\"http://www.example.com/sms/advanced\",
\"notifyContentType\":\"application/json\", \"callbackData\":\"DLR callback
data\", \"validityPeriod\": 720}, {\"from\":\"41793026700\",
\"destinations\":[{\"to\":\"41793026785\"}], \"text\":\"A long time ago, in
a galaxy far, far away... It is a period of civil war. Rebel spaceships,
striking from a hidden base, have won their first victory against the evil
Galactic Empire.\",\n  \t\t\t\"sendAt\":\"2015-07-
07T17:00:00.000+01:00\"}]}",  ParameterType.RequestBody);
IRestResponse response = client.Execute(request);
```

JavaScript

```
var data = JSON.stringify({
  "bulkId": "BULK-ID-123-xyz",
  "messages": [
    {
      "from": "InfoSMS",
      "destinations": [
        {
          "to": "41793026727",
          "messageId":  "MESSAGE-ID-123-xyz"
        },
        {
          "to": "41793026731"
        }
      ],
      "text": "Artık Ulusal Dil Tanımlayıcısı ile Türkçe karakterli
smslerinizi rahatlıkla iletebilirsiniz.",
      "flash": false,
      "language": {
        "languageCode": "TR",
        "singleShift": true,
        "lockingShift": false
      },
      "transliteration": "TURKISH",
      "notifyUrl": "http://www.example.com/sms/advanced",
      "notifyContentType": "application/json",
      "callbackData": "DLR callback data",
      "validityPeriod": 720
    },
    {
      "from": "41793026700",
      "destinations": [
        {
          "to": "41793026785"
        }
      ],
      "text": "A long time ago, in a galaxy far, far away... It is a period
of civil war. Rebel spaceships, striking from a hidden base, have won their
first victory against the evil Galactic Empire.",
      "sendAt":  "2015-07-07T17:00:00.000+01:00"
    }
  ]
});
```

```
var xhr = new XMLHttpRequest();
xhr.withCredentials = true;

xhr.addEventListener("readystatechange", function () {
  if (this.readyState === this.DONE) {
    console.log(this.responseText);
  }
});

xhr.open("POST", "http://107.20.199.106/restapi/sms/1/text/advanced");
xhr.setRequestHeader("content-type", "application/json");
xhr.setRequestHeader("authorization", "Basic
QWxhZGRpbjpvcGVuIHNlc2FtZQ==");
xhr.setRequestHeader("accept", "application/json");

xhr.send(data);
```

Result Format

JSON

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

{
   "bulkId": "BULK-ID-123-xyz",
   "messages":[
      {
         "to":"41793026727",
         "status":{
            "groupId":0,
            "groupName":"ACCEPTED",
            "id":0,
            "name":"MESSAGE_ACCEPTED",
            "description":"Message accepted"
         },
         "smsCount":1,
         "messageId":"MESSAGE-ID-123-xyz"
      },
      {
         "to":"41793026731",
         "status":{
            "groupId":0,
            "groupName":"ACCEPTED",
```

```
            "id":0,
            "name":"MESSAGE_ACCEPTED",
            "description":"Message accepted"
         },
         "smsCount":1,
         "messageId":"9304a5a3ab19-1ca1-be74-76ad87651ed25f35"
      },
      {
         "to":"41793026785",
         "status":{
            "groupId":0,
            "groupName":"ACCEPTED",
            "id":0,
            "name":"MESSAGE_ACCEPTED",
            "description":"Message accepted"
         },
         "smsCount":2,
         "messageId":"5f35f87a2f19-a141-43a4-91cd81b85f8c689"
      }
   ]
}
```


XML


```
HTTP/1.1 200 OK
Content-Type: application/xml

<SendSMSResponse>
   <bulkId>BULK-ID-123-xyz</bulkId>
   <messages>
      <messages>
         <to>41793026727</to>
         <status>
            <groupId>0</groupId>
            <groupName>ACCEPTED</groupName>
            <id>0</id>
            <name>MESSAGE_ACCEPTED</name>
            <description>Message  accepted</description>
         </status>
         <smsCount>1</smsCount>
         <messageId>MESSAGE-ID-123-xyz</messageId>
      </messages>
      <messages>
         <to>41793026731</to>
```

```
            <status>
               <groupId>0</groupId>
               <groupName>ACCEPTED</groupName>
               <id>0</id>
               <name>MESSAGE_ACCEPTED</name>
               <description>Message  accepted</description>
            </status>
            <smsCount>1</smsCount>
            <messageId>9304a5a3ab19-1ca1-be74-76ad87651ed25f35</messageId>
         </messages>
         <messages>
            <to>41793026785</to>
            <status>
               <groupId>0</groupId>
               <groupName>ACCEPTED</groupName>
               <id>0</id>
               <name>MESSAGE_ACCEPTED</name>
               <description>Message  accepted</description>
            </status>
            <smsCount>2</smsCount>
            <messageId>5f35f87a2f19-a141-43a4-91cd81b85f8c689</messageId>
         </messages>
      </messages>
   </SendSMSResponse>
```

Responseformat

If successful, response header HTTP status code will be `200 OK` and the message will be sent.

If you try to send the message without authorization, you will receive an `401 Unauthorized` error .

SMSResponse

Parameter	Type	Description
bulkId	String	The ID that uniquely identifies the request. Bulk ID will be received only when a message is sent to more than one destination address.
messages	SMSResponseDetails[]	Array of sent message objects, one object per every message.

SMSResponseDetails

Parameter	Type	Description
to	String	The message destination address.
status	Status	Indicates whether the message is sent successfully, not sent, delivered, not delivered, waiting for delivery or any other possible status.
smsCount	int	The number of sent message segments.
messageId	String	The ID that uniquely identifies the sent message.

Status

Parameter	Type	Description
groupId	int	Status group ID.
groupName	String	Status group name.
id	int	Status ID.
name	String	Status name.
description	String	Human readable description of the status.
action	String	Action that should be taken to eliminate the error.

Get account balance

This method allows you to get your account balance.

Definition

`http://107.20.199.106/restapi/account/1/balance`

Examples

JSON

```
GET /restapi/account/1/balance HTTP/1.1
Host: 107.20.199.106
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
```

```
Accept: application/json
```

XML

```
GET /restapi/account/1/balance HTTP/1.1
Host: 107.20.199.106
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
Accept: application/xml
```

cURL

```
curl -X GET \
-H 'Accept: application/json' \
-H "Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==" \
http://107.20.199.106/restapi/account/1/balance
```

PHP

```
<?php

$request = new HttpRequest();
$request->setUrl('http://107.20.199.106/restapi/account/1/balance');
$request->setMethod(HTTP_METH_GET);

$request->setHeaders(array(
  'accept' => 'application/json',
  'authorization' => 'Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ=='
));

try {
  $response = $request->send();

  echo $response->getBody();
} catch (HttpException $ex) {
  echo $ex;
}
```

Ruby

```
require 'uri'
```

```ruby
require 'net/http'

url = URI("http://107.20.199.106/restapi/account/1/balance")

http = Net::HTTP.new(url.host, url.port)
http.use_ssl = true
http.verify_mode = OpenSSL::SSL::VERIFY_NONE

request = Net::HTTP::Get.new(url)
request["authorization"] = 'Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ=='
request["accept"] = 'application/json'

response = http.request(request)
puts response.read_body
```

Python

```python
import http.client

conn = http.client.httpConnection("107.20.199.106")

headers = {
    'authorization': "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==",
    'accept': "application/json"
    }

conn.request("GET", "/restapi/account/1/balance", headers=headers)

res = conn.getresponse()
data = res.read()

print(data.decode("utf-8"))
```

Java

```java
HttpResponse<String> response =
Unirest.get("http://107.20.199.106/restapi/account/1/balance")
  .header("authorization", "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==")
  .header("accept",  "application/json")
  .asString();
```

C#

```
var client = new
RestClient("http://107.20.199.106/restapi/account/1/balance");

var request = new RestRequest(Method.GET);
request.AddHeader("accept",  "application/json");
request.AddHeader("authorization", "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==");

IRestResponse response = client.Execute(request);
```

JavaScript

```
var data = null;

var xhr = new XMLHttpRequest();
xhr.withCredentials = true;

xhr.addEventListener("readystatechange", function () {
  if (this.readyState === this.DONE) {
    console.log(this.responseText);
  }
});

xhr.open("GET", "http://107.20.199.106/restapi/account/1/balance");
xhr.setRequestHeader("authorization", "Basic
QWxhZGRpbjpvcGVuIHNlc2FtZQ==");
xhr.setRequestHeader("accept", "application/json");

xhr.send(data);
```

Result Format

JSON

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

{
  "balance": 47.79134,
  "currency": "EUR"
}
```

XML

```
HTTP/1.1 200 OK
Content-Type: application/xml

<AccountBalance>
    <balance>47.79134</balance>
    <currency>EUR</currency>
</AccountBalance>
```

Responseformat

On success, response header HTTP status code will be `200 OK` and delivery reports will be returned in the response body.

If you try to send a message without authorization, you will get a response with HTTP status code `401 Unauthorized`.

AccountBalance

Parameter	Type	Description
balance	Double	Account balance.
currency	String	The currency in which the account balance is expressed.

ReceiveSMS

Pull receivedmessages

This method allows you to pull received messages one time.

Definition

```
http://107.20.199.106/restapi/sms/1/inbox/reports
```

Parameters

Parameter	Type	Description
limit	Integer	Maximum number of received messages that will be returned.

Examples

```
GET /restapi/sms/1/inbox/reports HTTP/1.1
Host: 107.20.199.106
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
Accept: application/json
```

Result Format


```
{
   "results":[
      {
         "messageId":"ff4804ef-6ab6-4abd-984d-ab3b1387e823",
         "from":"38598111",
         "to":"41793026727",
         "text":"KEY Test message",
         "cleanText":"Test message",
         "keyword":"KEY",
         "receivedAt":"2015-02-15T11:43:20.254+0100",
         "smsCount":1
      }
   ]
}
```


> **Important:**
>
> In order to pull received messages, first you need a phone number and to **setup a pull action** on that number. Once you retrieve a received message, you will not be able to get the same message again by using this endpoint.

Responseformat

If successful, response header HTTP status code will be `200 OK` and messages will be returned in the response body.

If you try to get received messages without authorization, you will get a response with HTTP status code `401 Unauthorized` .

SMSResponse

Parameter	Type	Description
results	Messages[]	Collection of reports, one per every received message.

Messages

Parameter	Type	Description
messageId	String	The ID that uniquely identifies the received message.
from	String	Sender ID that can be alphanumeric or numeric.
to	String	The message destination address.
text	String	Full text of the received message.
cleanText	String	Text of received message without a keyword (if a keyword was sent).
keyword	String	Keyword extracted from the message text.
receivedAt	Date	Tells when the message was received on the platform. Has the following format: `yyyy-MM-dd'T'HH:mm:ss.SSSXXX` .
smsCount	int	The number of sent message segments.

Additional examples

Getting unread received messages without any query parameter

Request

JSON

```
GET /restapi/sms/1/inbox/reports HTTP/1.1
```

```
Host: 107.20.199.106
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
Accept: application/json
```

XML

```
GET /restapi/sms/1/inbox/reports HTTP/1.1
Host: 107.20.199.106
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
Accept: application/xml
```

cURL

```
curl -X GET \
-H 'Accept: application/json' \
-H "Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==" \
http://107.20.199.106/restapi/sms/1/inbox/reports
```

PHP

```
<?php

$request = new HttpRequest();
$request->setUrl('http://107.20.199.106/restapi/sms/1/inbox/reports');
$request->setMethod(HTTP_METH_GET);

$request->setHeaders(array(
  'accept' => 'application/json',
  'authorization' => 'Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ=='
));

try {
  $response = $request->send();

  echo $response->getBody();
} catch (HttpException $ex) {
  echo $ex;
}
```

Ruby

```ruby
require 'uri'
require 'net/http'

url = URI("http://107.20.199.106/restapi/sms/1/inbox/reports")

http = Net::HTTP.new(url.host, url.port)
http.use_ssl = true
http.verify_mode = OpenSSL::SSL::VERIFY_NONE

request = Net::HTTP::Get.new(url)
request["authorization"] = 'Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ=='
request["accept"] = 'application/json'

response = http.request(request)
puts response.read_body
```

Python

```python
import http.client

conn = http.client.httpConnection("107.20.199.106")

headers = {
    'authorization': "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==",
    'accept': "application/json"
    }

conn.request("GET", "/restapi/sms/1/inbox/reports", headers=headers)

res = conn.getresponse()
data = res.read()

print(data.decode("utf-8"))
```

Java

```java
HttpResponse<String> response =
Unirest.get("http://107.20.199.106/restapi/sms/1/inbox/reports")
  .header("authorization", "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==")
  .header("accept",  "application/json")
  .asString();
```

C#

```
var client = new
RestClient("http://107.20.199.106/restapi/sms/1/inbox/reports");

var request = new RestRequest(Method.GET);
request.AddHeader("accept",  "application/json");
request.AddHeader("authorization", "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==");

IRestResponse response = client.Execute(request);
```

JavaScript

```
var data = null;

var xhr = new XMLHttpRequest();
xhr.withCredentials = true;

xhr.addEventListener("readystatechange", function () {
  if (this.readyState === this.DONE) {
    console.log(this.responseText);
  }
});

xhr.open("GET", "http://107.20.199.106/restapi/sms/1/inbox/reports");
xhr.setRequestHeader("authorization", "Basic
QWxhZGRpbjpvcGVuIHNlc2FtZQ==");
xhr.setRequestHeader("accept", "application/json");

xhr.send(data);
```

Response

JSON

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

{
   "results":[
      {
         "messageId":"ff4804ef-6ab6-4abd-984d-ab3b1387e823",
```

```
            "from":"38598111",
            "to":"41793026727",
            "text":"KEY Test message",
            "cleanText":"Test message",
            "keyword":"KEY",
            "receivedAt":"2015-02-15T11:43:20.254+0100",
            "smsCount":1
        }
    ]
}
```

XML

```
HTTP/1.1 200 OK
Content-Type: application/xml

<MOReportResponse>
   <results>
      <results>
         <messageId>ff4804ef-6ab6-4abd-984d-ab3b1387e823</messageId>
         <from>38598111</from>
         <to>41793026727</to>
         <text>KEY Test message</text>
         <cleanText>Test  message</cleanText>
         <keyword>KEY</keyword>
         <receivedAt>2015-02-15T11:43:20.254+0100</receivedAt>
         <smsCount>1</smsCount>
      </results>
   </results>
</MOReportResponse>
```

Getting the initial two unread received messages

Request

JSON

```
GET /restapi/sms/1/inbox/reports?limit=2 HTTP/1.1
Host: 107.20.199.106
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
Accept: application/json
```

XML

```
GET /restapi/sms/1/inbox/reports?limit=2 HTTP/1.1
Host: 107.20.199.106
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
Accept: application/xml
```

cURL

```
curl -X GET
-H 'Accept: application/json'
-H "Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ=="
http://107.20.199.106/restapi/sms/1/inbox/reports?limit=2
```

PHP

```
<?php

$request = new HttpRequest();
$request->setUrl('http://107.20.199.106/restapi/sms/1/inbox/reports');
$request->setMethod(HTTP_METH_GET);

$request->setQueryData(array(
  'limit' => '2'
));

$request->setHeaders(array(
  'accept' => 'application/json',
  'authorization' => 'Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ=='
));

try {
  $response = $request->send();

  echo $response->getBody();
} catch (HttpException $ex) {
  echo $ex;
}
```

Ruby

```ruby
require 'uri'
require 'net/http'

url  =  URI("http://107.20.199.106/restapi/sms/1/inbox/reports?limit=2")

http = Net::HTTP.new(url.host, url.port)
http.use_ssl = true
http.verify_mode = OpenSSL::SSL::VERIFY_NONE

request =Net::HTTP::Get.new(url)
request["authorization"] = 'Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ=='
request["accept"] ='application/json'

response =http.request(request)
puts response.read_body
```

Python

```python
import http.client

conn = http.client.httpConnection("107.20.199.106")

headers = {
    'authorization': "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==",
    'accept': "application/json"
    }

conn.request("GET", "/restapi/sms/1/inbox/reports?limit=2",
headers=headers)

res = conn.getresponse()
data = res.read()

print(data.decode("utf-8"))
```

Java

```java
HttpResponse<String> response =
Unirest.get("http://107.20.199.106/restapi/sms/1/inbox/reports?limit=2")
  .header("authorization", "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==")
  .header("accept",  "application/json")
  .asString();
```

C#

```
var client = new
RestClient("http://107.20.199.106/restapi/sms/1/inbox/reports?limit=2");

var request = new RestRequest(Method.GET);
request.AddHeader("accept",  "application/json");
request.AddHeader("authorization", "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==");

IRestResponse response = client.Execute(request);
```

JavaScript

```
var data = null;

var xhr = new XMLHttpRequest();
xhr.withCredentials = true;

xhr.addEventListener("readystatechange", function () {
  if (this.readyState === this.DONE) {
    console.log(this.responseText);
  }
});

xhr.open("GET", "http://107.20.199.106/restapi/sms/1/inbox/reports?
limit=2");
xhr.setRequestHeader("authorization", "Basic
QWxhZGRpbjpvcGVuIHNlc2FtZQ==");
xhr.setRequestHeader("accept",  "application/json");

xhr.send(data);
```

Response

JSON

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

{
   "results":[
```

```
        {
            "messageId":"df9839ab-6ab6-4abd-984d-ab3b7687e823",
            "from":"38598111",
            "to":"41793026727",
            "text":"KEYWORD Test message",
            "cleanText":"Test message",
            "keyword":"KEYWORD",
            "receivedAt":"2015-02-15T12:23:10.418+0100",
            "smsCount":1
        },
        {
            "messageId":"ae4359fd-9ad6-4cdf-763d-ac1c5347d839",
            "from":"38598111",
            "to":"41793026727",
            "text":"Test message without keyword.",
            "cleanText":"Test message without keyword.",
            "keyword":"",
            "receivedAt":"2015-02-15T12:21:07.311+0100",
            "smsCount":1
        }
    ]
}
```


XML


```
HTTP/1.1 200 OK
Content-Type: application/xml

<MOReportResponse>
   <results>
      <results>
         <messageId>df9839ab-6ab6-4abd-984d-ab3b7687e823</messageId>
         <from>38598111</from>
         <to>41793026727</to>
         <text>KEYWORD Test message</text>
         <cleanText>Test  message</cleanText>
         <keyword>KEYWORD</keyword>
         <receivedAt>2015-02-15T12:23:10.418+0100</receivedAt>
         <smsCount>1</smsCount>
      </results>
      <results>
         <messageId>ae4359fd-9ad6-4cdf-763d-ac1c5347d839</messageId>
         <from>38598111</from>
         <to>41793026727</to>
```

```
            <text>Test message without keyword.</text>
            <cleanText>Test message without keyword.</cleanText>
            <keyword />
            <receivedAt>2015-02-15T12:21:07.311+0100</receivedAt>
            <smsCount>1</smsCount>
         </results>
      </results>
   </MOReportResponse>
```

Received messages logs

Definition

```
http://107.20.199.106/restapi/sms/1/inbox/logs
```

Parameters

Parameter	Type	Description
limit	Integer	Maximum number of messages in returned logs.
keyword	String	Keyword used in received messages
to	String	The message destination address.
receivedSince	DateFormat	Lower limit on date and time of sending SMS.
receivedUntil	DateFormat	Upper limit on date and time of sending SMS.

Examples

```
GET /restapi/sms/1/inbox/logs HTTP/1.1
Host: 107.20.199.106
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
Accept: application/json
```

Result Format

```
{
   "results":[
      {
         "messageId":"5908971644001839114",
         "to":"41793026727",
         "receivedAt":"2015-03-01T12:54:44.560+0000",
         "from":"385998779111",
         "text":"HEY hello world",
         "cleanText":"hello world",
         "keyword":"HEY",
         "smsCount":1
      },
      {
         "messageId":"5904932597450690569",
         "to":"41793026727",
         "receivedAt":"2015-03-01T12:54:42.231+0000",
         "from":"385998779111",
         "text":"HEY how are you",
         "cleanText":"how are you",
         "keyword":"HEY",
         "smsCount":1
      },
      {
         "messageId":"5904217701796992008",
         "to":"41793026727",
         "receivedAt":"2015-03-01T12:54:40.111+0000",
         "from":"385998779111",
         "text":"KEY hello world",
         "cleanText":"hello world",
         "keyword":"KEY",
         "smsCount":1
      }
   ]
}
```


Responseformat

If successful, response header HTTP status code will be `200 OK` and the message logs will be returned.

If you try to send message without authorization, you will get a response with HTTP status code `401 Unauthorized`.

LogsResponse

Parameter	Type	Description
results	MOLog[]	Collection of logs.

MOLog

Parameter	Type	Description
messageId	String	The ID that uniquely identifies the received message.
to	String	The message destination address.
receivedAt	Date	Tells when the SMS was received. Has the following format: `yyyy-MM-dd'T'HH:mm:ss.SSSZ` .
from	String	Sender ID that can be alphanumeric or numeric.
text	String	Text of the message that was sent.
cleanText	String	Text of the message that was sent without the keyword.
keyword	String	Keyword extracted from the message text.
smsCount	int	The number of sent message segments.

Additional examples

Getting logs without any query parameter

Request

JSON

```
GET /restapi/sms/1/inbox/logs HTTP/1.1
Host: 107.20.199.106
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
Accept: application/json
```

XML

```
GET /restapi/sms/1/inbox/logs HTTP/1.1
Host: 107.20.199.106
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
Accept: application/xml
```

cURL

```
curl -X GET \
-H 'Accept: application/json' \
-H 'Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==' \
http://107.20.199.106/restapi/sms/1/inbox/logs
```

PHP

```
<?php

$request = new HttpRequest();
$request->setUrl('http://107.20.199.106/restapi/sms/1/inbox/logs');
$request->setMethod(HTTP_METH_GET);

$request->setHeaders(array(
  'accept' => 'application/json',
  'authorization' => 'Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ=='
));

try {
  $response = $request->send();

  echo $response->getBody();
} catch (HttpException $ex) {
  echo $ex;
}
```

Ruby

```
require 'uri'
require 'net/http'

url = URI("http://107.20.199.106/restapi/sms/1/inbox/logs")

http = Net::HTTP.new(url.host, url.port)
```

```
http.use_ssl = true
http.verify_mode = OpenSSL::SSL::VERIFY_NONE

request = Net::HTTP::Get.new(url)
request["authorization"] = 'Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ=='
request["accept"] = 'application/json'

response = http.request(request)
puts response.read_body
```

Python

```
import http.client

conn = http.client.httpConnection("107.20.199.106")

headers = {
    'authorization': "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==",
    'accept': "application/json"
    }

conn.request("GET", "/restapi/sms/1/inbox/logs", headers=headers)

res = conn.getresponse()
data = res.read()

print(data.decode("utf-8"))
```

Java

```
HttpResponse<String> response =
Unirest.get("http://107.20.199.106/restapi/sms/1/inbox/logs")
  .header("authorization", "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==")
  .header("accept",  "application/json")
  .asString();
```

C#

```
var client = new
RestClient("http://107.20.199.106/restapi/sms/1/inbox/logs");

var request = new RestRequest(Method.GET);
```

```
request.AddHeader("accept",  "application/json");
request.AddHeader("authorization", "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==");

IRestResponse response = client.Execute(request);
```

JavaScript

```
var data = null;

var xhr = new XMLHttpRequest();
xhr.withCredentials = true;

xhr.addEventListener("readystatechange", function () {
  if (this.readyState === this.DONE) {
    console.log(this.responseText);
  }
});

xhr.open("GET", "http://107.20.199.106/restapi/sms/1/inbox/logs");
xhr.setRequestHeader("authorization", "Basic
QWxhZGRpbjpvcGVuIHNlc2FtZQ==");
xhr.setRequestHeader("accept", "application/json");

xhr.send(data);
```

Response

JSON

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

{
   "results":[
      {
         "messageId":"5908971644001839114",
         "to":"41793026727",
         "receivedAt":"2015-03-01T12:54:44.560+0000",
         "from":"385998779111",
         "text":"HEY hello world",
         "cleanText":"hello world",
         "keyword":"HEY",
         "smsCount":1
```

```
        },
        {
            "messageId":"5904932597450690569",
            "to":"41793026727",
            "receivedAt":"2015-03-01T12:54:42.231+0000",
            "from":"385998779111",
            "text":"HEY how are you",
            "cleanText":"how are you",
            "keyword":"HEY",
            "smsCount":1
        },
        {
            "messageId":"5904217701796992008",
            "to":"41793026727",
            "receivedAt":"2015-03-01T12:54:40.111+0000",
            "from":"385998779111",
            "text":"KEY hello world",
            "cleanText":"hello world",
            "keyword":"KEY",
            "smsCount":1
        }
    ]
}
```


XML


```
HTTP/1.1 200 OK
Content-Type: application/xml

<MOLogsResponse>
   <results>
      <results>
         <messageId>590897164400183914</messageId>
         <to>13372</to>
         <receivedAt>2015-03-01T12:54:44.560+0000</receivedAt>
         <from>385998779613</from>
         <text>HEY hello world</text>
         <cleanText>hello  world</cleanText>
         <keyword>HEY</keyword>
         <smsCount>1</smsCount>
      </results>
      <results>
         <messageId>5904932597450690569</messageId>
         <to>13372</to>
```

```
            <receivedAt>2015-03-01T12:54:42.231+0000</receivedAt>
            <from>385998779613</from>
            <text>HEY how are you</text>
            <cleanText>how are you</cleanText>
            <keyword>HEY</keyword>
            <smsCount>1</smsCount>
         </results>
         <results>
            <messageId>5904217701796992008</messageId>
            <to>13372</to>
            <receivedAt>2015-03-01T12:54:40.111+0000</receivedAt>
            <from>385998779613</from>
            <text>KEY hello world</text>
            <cleanText>hello  world</cleanText>
            <keyword>KEY</keyword>
            <smsCount>1</smsCount>
         </results>
      </results>
   </MOLogsResponse>
```

Getting logs with `keyword` and `to` as filters

Request

JSON

```
GET /restapi/sms/1/inbox/logs?keyword=HEY&to=13372 HTTP/1.1
Host: 107.20.199.106
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
Accept: application/json
```

XML

```
GET /restapi/sms/1/inbox/logs?keyword=HEY&to=13372 HTTP/1.1
Host: 107.20.199.106
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
Accept: application/xml
```

cURL

```
curl -X GET
```

```
-H 'Accept: application/json'
-H 'Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ=='
http://107.20.199.106/restapi/sms/1/inbox/logs?keyword=HEY&to=13372
```

PHP

```
<?php

$request = new HttpRequest();
$request->setUrl('http://107.20.199.106/restapi/sms/1/inbox/logs');
$request->setMethod(HTTP_METH_GET);

$request->setQueryData(array(
  'keyword' => 'HEY',
  'to' => '13372'
));

$request->setHeaders(array(
  'accept' => 'application/json',
  'authorization' => 'Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ=='
));

try {
  $response = $request->send();

  echo $response->getBody();
} catch (HttpException $ex) {
  echo $ex;
}
```

Ruby

```
require 'uri'
require 'net/http'

url = URI("http://107.20.199.106/restapi/sms/1/inbox/logs?
keyword=HEY&to=13372")

http = Net::HTTP.new(url.host, url.port)
http.use_ssl = true
http.verify_mode = OpenSSL::SSL::VERIFY_NONE

request = Net::HTTP::Get.new(url)
```

```
request["authorization"] = 'Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ=='
request["accept"] ='application/json'

response =http.request(request)
puts response.read_body
```

Python

```
import http.client

conn = http.client.httpConnection("107.20.199.106")

headers = {
    'authorization': "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==",
    'accept': "application/json"
    }

conn.request("GET", "/restapi/sms/1/inbox/logs?keyword=HEY&to=13372",
headers=headers)

res = conn.getresponse()
data = res.read()

print(data.decode("utf-8"))
```

Java

```
HttpResponse<String> response =
Unirest.get("http://107.20.199.106/restapi/sms/1/inbox/logs?
keyword=HEY&to=13372")
  .header("authorization", "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==")
  .header("accept",  "application/json")
  .asString();
```

C#

```
var client = new
RestClient("http://107.20.199.106/restapi/sms/1/inbox/logs?
keyword=HEY&to=13372");

var request = new RestRequest(Method.GET);
request.AddHeader("accept",  "application/json");
```

```
request.AddHeader("authorization", "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==");

IRestResponse response = client.Execute(request);
```

JavaScript

```
var data = null;

var xhr = new XMLHttpRequest();
xhr.withCredentials = true;

xhr.addEventListener("readystatechange", function () {
  if (this.readyState === this.DONE) {
    console.log(this.responseText);
  }
});

xhr.open("GET", "http://107.20.199.106/restapi/sms/1/inbox/logs?
keyword=HEY&to=13372");
xhr.setRequestHeader("authorization", "Basic
QWxhZGRpbjpvcGVuIHNlc2FtZQ==");
xhr.setRequestHeader("accept",  "application/json");

xhr.send(data);
```

Response

JSON

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

{
   "results":[
      {
         "messageId":"5908971644001839114",
         "to":"41793026727",
         "receivedAt":"2015-03-01T12:54:44.560+0000",
         "from":"385998779111",
         "text":"HEY hello world",
         "cleanText":"hello world",
         "keyword":"HEY",
         "smsCount":1
```

```
        },
        {
            "messageId":"5904932597450690569",
            "to":"41793026727",
            "receivedAt":"2015-03-01T12:54:42.231+0000",
            "from":"385998779111",
            "text":"HEY how are you",
            "cleanText":"how are you",
            "keyword":"HEY",
            "smsCount":1
        }
    ]
}
```

XML

```
HTTP/1.1 200 OK
Content-Type: application/xml

<MOLogsResponse>
   <results>
      <results>
         <messageId>5908971644001839114</messageId>
         <to>13372</to>
         <receivedAt>2015-03-01T12:54:44.560+0000</receivedAt>
         <from>385998779613</from>
         <text>HEY hello world</text>
         <cleanText>hello  world</cleanText>
         <keyword>HEY</keyword>
         <smsCount>1</smsCount>
      </results>
      <results>
         <messageId>5904932597450690569</messageId>
         <to>13372</to>
         <receivedAt>2015-03-01T12:54:42.231+0000</receivedAt>
         <from>385998779613</from>
         <text>HEY how are you</text>
         <cleanText>how are you</cleanText>
         <keyword>HEY</keyword>
         <smsCount>1</smsCount>
      </results>
   </results>
</MOLogsResponse>
```

Getting messages without keyword

In order to get all messages that were sent without keyword, you can use `NO_KEYWORD` filter.

Request

JSON

```
GET /restapi/sms/1/inbox/logs?keyword=NO_KEYWORD HTTP/1.1
Host: 107.20.199.106
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
Accept: application/json
```

XML

```
GET /restapi/sms/1/inbox/logs?keyword=NO_KEYWORD HTTP/1.1
Host: 107.20.199.106
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
Accept: application/xml
```

cURL

```
curl -X GET
-H 'Accept: application/json'
-H 'Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ=='
http://107.20.199.106/restapi/sms/1/inbox/logs?keyword=NO_KEYWORD
```

PHP

```
<?php

$request = new HttpRequest();
$request->setUrl('http://107.20.199.106/restapi/sms/1/inbox/logs');
$request->setMethod(HTTP_METH_GET);

$request->setQueryData(array(
  'keyword' => 'NO_KEYWORD'
));

$request->setHeaders(array(
  'accept' => 'application/json',
```

```
    'authorization' => 'Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ=='
));

try {
  $response = $request->send();

  echo $response->getBody();
} catch (HttpException $ex) {
  echo $ex;
}
```

Ruby

```
require 'uri'
require 'net/http'

url = URI("http://107.20.199.106/restapi/sms/1/inbox/logs?
keyword=NO_KEYWORD")

http = Net::HTTP.new(url.host, url.port)
http.use_ssl = true
http.verify_mode = OpenSSL::SSL::VERIFY_NONE

request = Net::HTTP::Get.new(url)
request["authorization"] = 'Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ=='
request["accept"] = 'application/json'

response = http.request(request)
puts response.read_body
```

Python

```
import http.client

conn = http.client.HTTPConnection("107.20.199.106")

headers = {
    'authorization': "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==",
    'accept': "application/json"
    }

conn.request("GET", "/restapi/sms/1/inbox/logs?keyword=NO_KEYWORD",
headers=headers)
```

```
res = conn.getresponse()
data = res.read()

print(data.decode("utf-8"))
```

Java

```
HttpResponse<String> response =
Unirest.get("http://107.20.199.106/restapi/sms/1/inbox/logs?
keyword=NO_KEYWORD")
  .header("authorization", "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==")
  .header("accept",  "application/json")
  .asString();
```

C#

```
var client = new
RestClient("http://107.20.199.106/restapi/sms/1/inbox/logs?
keyword=NO_KEYWORD");

var request = new RestRequest(Method.GET);
request.AddHeader("accept",  "application/json");
request.AddHeader("authorization", "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==");

IRestResponse response = client.Execute(request);
```

JavaScript

```
var data = null;

var xhr = new XMLHttpRequest();
xhr.withCredentials = true;

xhr.addEventListener("readystatechange", function () {
  if (this.readyState === this.DONE) {
    console.log(this.responseText);
  }
});

xhr.open("GET", "http://107.20.199.106/restapi/sms/1/inbox/logs?
keyword=NO_KEYWORD");
```

```
xhr.setRequestHeader("authorization", "Basic
QWxhZGRpbjpvcGVuIHNlc2FtZQ==");
xhr.setRequestHeader("accept",  "application/json");

xhr.send(data);
```

Response

JSON

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

{
   "results":[
      {
         "messageId":"606386740726357762",
         "from":"385998779111",
         "to":"41793026727",
         "text":"Message without keyword",
         "cleanText":"Message without keyword",
         "receivedAt":"2015-05-20T10:06:46.880+0000",
         "smsCount":1
      },
      {
         "messageId":"3634590217319761028",
         "from":"385998779111",
         "to":"41793026727",
         "text":"Logs test",
         "cleanText":"Logs test",
         "receivedAt":"2015-05-20T10:06:17.713+0000",
         "smsCount":1
      }
   ]
}
```

XML

```
HTTP/1.1 200 OK
Content-Type: application/xml

<MOLogsResponse>
    <results>
```

```
        <messageId>60638674072635776</messageId>
        <from>385998779111</from>
        <to>41793026727</to>
        <text>Message without keyword</text>
        <cleanText>Message without keyword</cleanText>
        <receivedAt>2015-05-20T10:06:46.880+0000</receivedAt>
        <smsCount>1</smsCount>
    </results>
    <results>
        <messageId>36345902173197610 28</messageId>
        <from>385998779111</from>
        <to>41793026727</to>
        <text>Logs test</text>
        <cleanText>Logs  test</cleanText>
        <receivedAt>2015-05-20T10:06:17.713+0000</receivedAt>
        <smsCount>1</smsCount>
    </results>
</MOLogsResponse>
```

Number Context

Synchronous request

This method gives you the ability to make a synchronous Number Context request over HTTP. Number Context response is returned immediately thus eliminating the need for the call back server.

Definition

```
http://107.20.199.106/restapi/number/1/query
```

Parameters

Parameter	Type	Description
to	String[]	**Required.** Array of Number Context destination addresses. If the Number Context lookup is requested for one phone number, a single String is supported

instead of an Array. Destination addresses must be in international format (Example: `41793026727`).

Examples

```
POST /restapi/number/1/query HTTP/1.1
Host: 107.20.199.106
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
Content-Type: application/json
Accept: application/json

{
   "to":["41793026727"]
}
```

Result Format


```
{
   "results":[
      {
         "to":"41793026727",
         "mccMnc":"22801",
         "originalNetwork":{
            "networkPrefix":"79",
            "countryPrefix":"41"
         },
         "ported":false,
         "roaming":false,
         "status":{
            "groupId":2,
            "groupName":"UNDELIVERABLE",
            "id":9,
            "name":"UNDELIVERABLE_NOT_DELIVERED",
            "description":"Message sent not delivered"
         },
         "error":{
            "groupId":1,
            "groupName":"HANDSET_ERRORS",
            "id":27,
            "name":"EC_ABSENT_SUBSCRIBER",
            "description":"Absent  Subscriber",
```

```
            "permanent":false
        }
    }
  ]
}
```

Responseformat

If successful, response header HTTP status code will be `200 OK` and include Number Context information in responsebody.

If you try to send a Number Context lookup without authorization, you will get a response with HTTP status code `401 Unauthorized`.

NCResponse

Parameter	Type	Description
bulkId*	String	The ID that uniquely identifies the request. Bulk ID will be received only when you send a Number Context lookup to more than one destination address.
results	NCResponseDetails[]	Array of Number Context lookup results, one per every phone number.

NCResponseDetails

Parameter	Type	Description
to	String	The Number Context destination address.
mccMnc	String	Mobile country code and mobile network code concatenated. MccMnc will start with the MCC, and it will always have three digits, followed by the MNC (length of the MNC depends on the value of the MCC, and it can be two or three).
imsi	String	International Mobile Subscriber Identity, used to uniquely identify the user of a mobile network.
originalNetwork	Network	Information about the original network.
ported	Boolean	Tells if the phone number is ported.

portedNetwork	Network	Information about the ported network.
roaming	Boolean	Informs if the phone number is in roaming.
roamingNetwork	Network	Information about the roaming network.
servingMSC	String	Serving mobile switching center.
status	Status	Indicates whether the Number Context query was successfully executed, not executed or any other possible status.
error	Error	Indicates whether the error occurred during the query execution.

Network

Parameter	Type	Description
networkName	String	Network name.
networkPrefix	String	Network prefix.
countryName	String	Country name.
countryPrefix	String	Country prefix.

Status

Parameter	Type	Description
groupId	int	Status group ID.
groupName	String	Status group name.
id	int	Status ID.
name	String	Status name.
description	String	Human readable description of the status.
action	String	Action that should be taken to eliminate the error.

Error

Parameter	Type	Description
groupId	int	Error group ID.

groupName	String	Error group name.
id	int	Error ID.
name	String	Error name.
description	String	Human readable description of the error.
permanent	boolean	Tells if the error is permanent.

Additional examples

Number Context lookup - Single phone number

Request

JSON

```
POST /restapi/number/1/query HTTP/1.1
Host: 107.20.199.106
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
Content-Type: application/json
Accept: application/json

{
   "to":["41793026727"]
}
```

XML

```
POST /restapi/number/1/query HTTP/1.1
Host: 107.20.199.106
Authorization: Basic dXNlcm5hbWU6cGFzc3dvcmQ=
Content-Type: application/xml
Accept: application/xml

<request>
    <to>41793026727</to>
</request>
```

cURL

```
curl -X POST \
-H "Content-Type: application/json" \
-H 'Accept: application/json' \
-H "Authorization: Basic dXNlcm5hbWU6cGFzc3dvcmQ=" \
-d '{
   "to":["41793026727"]
}'  http://107.20.199.106/restapi/number/1/query
```

PHP

```
<?php

$request = new HttpRequest();
$request->setUrl('http://107.20.199.106/restapi/number/1/query');
$request->setMethod(HTTP_METH_POST);

$request->setHeaders(array(
  'content-type' => 'application/json',
  'accept' => 'application/json',
  'authorization' => 'Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ=='
));

$request->setBody('{
   "to":["41793026727"]
}');

try {
  $response = $request->send();

  echo $response->getBody();
} catch (HttpException $ex) {
  echo $ex;
}
```

Ruby

```
require 'uri'
require 'net/http'

url = URI("http://107.20.199.106/restapi/number/1/query")
```

```
http = Net::HTTP.new(url.host, url.port)
http.use_ssl = true
http.verify_mode = OpenSSL::SSL::VERIFY_NONE

request = Net::HTTP::Post.new(url)
request["authorization"] = 'Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ=='
request["accept"] = 'application/json'
request["content-type"] = 'application/json'

request.body = "{\"to\":[\"41793026727\"]}"

response = http.request(request)
puts response.read_body
```

Python

```
import http.client

conn = http.client.HTTPConnection("107.20.199.106")

payload = "{\"to\":[\"41793026727\"]}"

headers = {
    'authorization': "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==",
    'accept': "application/json",
    'content-type':  "application/json"
    }

conn.request("POST", "/restapi/number/1/query", payload, headers)

res = conn.getresponse()
data = res.read()

print(data.decode("utf-8"))
```

Java

```
HttpResponse<String> response =
Unirest.post("http://107.20.199.106/restapi/number/1/query")
  .header("authorization", "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==")
  .header("accept",  "application/json")
  .header("content-type",  "application/json")
```

```
  .body("{\"to\":[\"41793026727\"]}")
  .asString();
```

C#

```
var client = new
RestClient("http://107.20.199.106/restapi/number/1/query");

var request = new RestRequest(Method.POST);
request.AddHeader("content-type", "application/json");
request.AddHeader("accept",  "application/json");
request.AddHeader("authorization", "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==");
request.AddParameter("application/json", "{\"to\":[\"41793026727\"]}",
ParameterType.RequestBody);

IRestResponse response = client.Execute(request);
```

JavaScript

```
var data = JSON.stringify({
  "to": [
    "41793026727"
  ]
});

var xhr = new XMLHttpRequest();
xhr.withCredentials = true;

xhr.addEventListener("readystatechange", function () {
  if (this.readyState === this.DONE) {
    console.log(this.responseText);
  }
});

xhr.open("POST", "http://107.20.199.106/restapi/number/1/query");
xhr.setRequestHeader("authorization", "Basic
QWxhZGRpbjpvcGVuIHNlc2FtZQ==");
xhr.setRequestHeader("accept", "application/json");
xhr.setRequestHeader("content-type",   "application/json");

xhr.send(data);
```

Response

JSON

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

{
   "results":[
      {
         "to":"41793026727",
         "mccMnc":"22801",
         "originalNetwork":{
            "networkPrefix":"79",
            "countryPrefix":"41"
         },
         "ported":false,
         "roaming":false,
         "status":{
            "groupId":2,
            "groupName":"UNDELIVERABLE",
            "id":9,
            "name":"UNDELIVERABLE_NOT_DELIVERED",
            "description":"Message sent not delivered"
         },
         "error":{
            "groupId":1,
            "groupName":"HANDSET_ERRORS",
            "id":27,
            "name":"EC_ABSENT_SUBSCRIBER",
            "description":"Absent Subscriber",
            "permanent":false
         }
      }
   ]
}
```

XML

```
HTTP/1.1 200 OK
Content-Type: application/xml

<response>
   <ncResults>
```

```
      <ncResults>
         <to>41793026727</to>
         <mccMnc>22801</mccMnc>
         <originalNetwork>
            <networkPrefix>79</networkPrefix>
            <countryPrefix>41</countryPrefix>
         </originalNetwork>
         <ported>false</ported>
         <roaming>false</roaming>
         <status>
            <groupId>2</groupId>
            <groupName>UNDELIVERABLE</groupName>
            <id>9</id>
            <name>UNDELIVERABLE_NOT_DELIVERED</name>
            <description>Message sent not delivered</description>
         </status>
         <error>
            <groupId>1</groupId>
            <groupName>HANDSET_ERRORS</groupName>
            <id>27</id>
            <name>EC_ABSENT_SUBSCRIBER</name>
            <description>Absent  Subscriber</description>
            <permanent>false</permanent>
         </error>
      </ncResults>
   </ncResults>
</NCResponse>
```

Number Context lookup - Multiple phone numbers

Request

JSON

```
POST /restapi/number/1/query HTTP/1.1
Host: 107.20.199.106
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
Content-Type: application/json
Accept: application/json

{
   "to":[
      "41793026727",
```

```
      "3859851212"
   ]
}
```

XML

```
POST /restapi/number/1/query HTTP/1.1
Host: 107.20.199.106
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
Content-Type: application/xml
Accept: application/xml

<request>
   <to>
      <to>41793026727</to>
      <to>3859851212</to>
   </to>
</request>
```

cURL

```
curl -X POST
-H "Content-Type: application/json"
-H 'Accept: application/json'
-H "Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ="
-d '{
   "to":[
      "41793026727",
      "3859851212"
   ]
}'  http://107.20.199.106/restapi/number/1/query
```

PHP

```
<?php

$request = new HttpRequest();
$request->setUrl('http://107.20.199.106/restapi/number/1/query');
$request->setMethod(HTTP_METH_POST);

$request->setHeaders(array(
  'content-type' => 'application/json',
```

```
    'accept' => 'application/json',
    'authorization' => 'Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ=='
));

$request->setBody('{
   "to":[
      "41793026727",
      "3859851212"
   ]
}');

try {
  $response = $request->send();

  echo $response->getBody();
} catch (HttpException $ex) {
  echo $ex;
}
```

Ruby

```
require 'uri'
require 'net/http'

url = URI("http://107.20.199.106/restapi/number/1/query")

http = Net::HTTP.new(url.host, url.port)
http.use_ssl = true
http.verify_mode = OpenSSL::SSL::VERIFY_NONE

request = Net::HTTP::Post.new(url)
request["authorization"] = 'Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ=='
request["accept"] = 'application/json'
request["content-type"] = 'application/json'

request.body = "{\"to\":[\"41793026727\", \"3859851212\"]}"

response = http.request(request)
puts response.read_body
```

Python

```
import http.client
```

```
conn = http.client.httpConnection("107.20.199.106")

payload = "{\"to\":[\"41793026727\", \"3859851212\"]}"

headers = {
    'authorization': "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==",
    'accept': "application/json",
    'content-type':  "application/json"
    }

conn.request("POST", "/restapi/number/1/query", payload, headers)

res = conn.getresponse()
data = res.read()

print(data.decode("utf-8"))
```

Java

```
HttpResponse<String> response =
Unirest.post("http://107.20.199.106/restapi/number/1/query")
  .header("authorization", "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==")
  .header("accept",  "application/json")
  .header("content-type",  "application/json")
  .body("{\"to\":[\"41793026727\",   \"3859851212\"]}")
  .asString();
```

C#

```
var client = new
RestClient("http://107.20.199.106/restapi/number/1/query");

var request = new RestRequest(Method.POST);
request.AddHeader("content-type", "application/json");
request.AddHeader("accept",  "application/json");
request.AddHeader("authorization",  "Basic  QWxhZGRpbjpvcGVuIHNlc2FtZQ==");
request.AddParameter("application/json",   "{\"to\":[\"41793026727\",
\"3859851212\"]}", ParameterType.RequestBody);

IRestResponse response = client.Execute(request);
```

JavaScript

```
var data = JSON.stringify({
  "to": [
    "41793026727",
    "3859851212"
  ]
});

var xhr = new XMLHttpRequest();
xhr.withCredentials = true;

xhr.addEventListener("readystatechange", function () {
  if (this.readyState === this.DONE) {
    console.log(this.responseText);
  }
});

xhr.open("POST", "http://107.20.199.106/restapi/number/1/query");
xhr.setRequestHeader("authorization", "Basic
QWxhZGRpbjpvcGVuIHNlc2FtZQ==");
xhr.setRequestHeader("accept", "application/json");
xhr.setRequestHeader("content-type",  "application/json");

xhr.send(data);
```

Response

JSON

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

{
   "bulkId":"f5c4322c-10e7-a41e-5528-34fat43er4134",
   "results":[
      {
         "to":"3859851212",
         "mccMnc":"21901",
         "originalNetwork":{
            "networkPrefix":"98",
            "countryPrefix":"385"
         },
         "ported":false,
```

```
            "roaming":false,
            "status":{
               "groupId":2,
               "groupName":"UNDELIVERABLE",
               "id":9,
               "name":"UNDELIVERABLE_NOT_DELIVERED",
               "description":"Message sent not delivered"
            },
            "error":{
               "groupId":1,
               "groupName":"HANDSET_ERRORS",
               "id":1,
               "name":"EC_UNKNOWN_SUBSCRIBER",
               "description":"Unknown Subscriber",
               "permanent":true
            }
         },
         {
            "to":"41793026727",
            "mccMnc":"22801",
            "originalNetwork":{
               "networkPrefix":"79",
               "countryPrefix":"41"
            },
            "ported":false,
            "roaming":false,
            "status":{
               "groupId":2,
               "groupName":"UNDELIVERABLE",
               "id":9,
               "name":"UNDELIVERABLE_NOT_DELIVERED",
               "description":"Message sent not delivered"
            },
            "error":{
               "groupId":1,
               "groupName":"HANDSET_ERRORS",
               "id":27,
               "name":"EC_ABSENT_SUBSCRIBER",
               "description":"Absent Subscriber",
               "permanent":false
            }
         }
      ]
   }
```

XML

```
HTTP/1.1 200 OK
Content-Type: application/xml

<NCResponse>
   <results>
      <bulkId>f5c4322c-10e7-a41e-5528-34fat43er4134</bulkId>
      <results>
         <to>41793026727</to>
         <mccMnc>22801</mccMnc>
         <originalNetwork>
            <networkPrefix>79</networkPrefix>
            <countryPrefix>41</countryPrefix>
         </originalNetwork>
         <ported>false</ported>
         <roaming>false</roaming>
         <status>
            <groupId>2</groupId>
            <groupName>UNDELIVERABLE</groupName>
            <id>9</id>
            <name>UNDELIVERABLE_NOT_DELIVERED</name>
            <description>Message sent not delivered</description>
         </status>
         <error>
            <groupId>1</groupId>
            <groupName>HANDSET_ERRORS</groupName>
            <id>27</id>
            <name>EC_ABSENT_SUBSCRIBER</name>
            <description>Absent  Subscriber</description>
            <permanent>false</permanent>
         </error>
      </ncResults>
      <ncResults>
         <to>3859851212</to>
         <mccMnc>21901</mccMnc>
         <originalNetwork>
            <networkPrefix>98</networkPrefix>
            <countryPrefix>385</countryPrefix>
         </originalNetwork>
         <ported>false</ported>
         <roaming>false</roaming>
         <status>
            <groupId>2</groupId>
            <groupName>UNDELIVERABLE</groupName>
```

```
            <id>9</id>
            <name>UNDELIVERABLE_NOT_DELIVERED</name>
            <description>Message sent not delivered</description>
         </status>
         <error>
            <groupId>1</groupId>
            <groupName>HANDSET_ERRORS</groupName>
            <id>1</id>
            <name>EC_UNKNOWN_SUBSCRIBER</name>
            <description>Unknown  Subscriber</description>
            <permanent>true</permanent>
         </error>
      </ncResults>
   </ncResults>
</NCResponse>
```

Asynchronous request

This method gives you the ability to make a asynchronous Number Context request over HTTP. Number Context response is sent to a notify URL at your call back server.

Definition

```
http://107.20.199.106/restapi/number/1/notify
```

Parameters

Parameter	Type	Description
to	String[]	**Required.** Array of Number Context destination addresses. If the Number Context lookup is requested for one phone number, a single String is supported instead of an Array. Destination addresses must be in international format (Example: `41793026727`).
notifyUrl	String	The URL on your call back server on which the Number Context response will be sent.
notifyContentType	String	Preferred content type of the response. Could be `application/json` or `application/xml` .

Examples

```
POST /restapi/number/1/notify HTTP/1.1
Host: 107.20.199.106
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
Content-Type: application/json
Accept: application/json

{
   "to":[
      "41793026727"
   ],
   "notifyUrl":"http://example.com/notifyUrl",
   "notifyContentType":"application/json"
}
```

Result Format

```
{
   "ncResults":[
      {
         "to":"41793026727",
         "status":{
            "groupId":1,
            "groupName":"PENDING",
            "id":3,
            "name":"PENDING_WAITING_DELIVERY",
            "description":"Message sent, waiting for delivery report"
         },
         "messageId":"2b691c32-1233-4716-a763-4f70cc929eae"
      }
   ]
}
```

Responseformat

If successful, response header HTTP status code will be `200 OK` and include Number Context information in the response body.

If you try to send a Number Context lookup without authorization, you will get a response with HTTP status code `401 Unauthorized`.

NCResponse

Parameter	Type	Description
bulkId	String	The ID that uniquely identifies the request. Bulk ID will be received only when you send a Number Context lookup to more than one destination address.
ncResults	NCResponseDetails[]	Array of Number Context lookup results, one per every phone number.

NCResponseDetails

Parameter	Type	Description
to	String	The Number Context destination address.
status	Status	Indicates whether the Number Context query was successfully executed, not executed or any other possible status.
messageId	String	The ID that uniquely identifies the performed lookup on destination address.

Status

Parameter	Type	Description
groupId	int	Status group ID.
groupName	String	Status group name.
id	int	Status ID.
name	String	Status name.
description	String	Human readable description of the status.
action	String	Action that should be taken to eliminate the error.

Error

Parameter	Type	Description

groupId	int	Error group ID.
groupName	String	Error group name.
id	int	Error ID.
name	String	Error name.
description	String	Human readable description of the error.
permanent	boolean	Tells if the error is permanent.

> **Note:**
>
> The result you will receive on your notify URL will be the same as the response in Synchronous Number Context lookup.

The result format you will receive on notify URL:

JSON


```
{
   "ncResults":[
      {
         "to":"41793026727",
         "mccMnc":"22801",
         "originalNetwork":{
            "networkPrefix":"79",
            "countryPrefix":"41"
         },
         "ported":false,
         "roaming":false,
         "status":{
            "groupId":2,
            "groupName":"UNDELIVERABLE",
            "id":9,
            "name":"UNDELIVERABLE_NOT_DELIVERED",
            "description":"Message sent not delivered"
         },
         "error":{
            "groupId":1,
            "groupName":"HANDSET_ERRORS",
            "id":27,
            "name":"EC_ABSENT_SUBSCRIBER",
```

```
            "description":"Absent Subscriber",
            "permanent":false
         }
      }
   ]
}
```

XML

```
<NCResponse>
   <ncResults>
      <ncResults>
         <to>41793026727</to>
         <mccMnc>22801</mccMnc>
         <originalNetwork>
            <networkPrefix>79</networkPrefix>
            <countryPrefix>41</countryPrefix>
         </originalNetwork>
         <ported>false</ported>
         <roaming>false</roaming>
         <status>
            <groupId>2</groupId>
            <groupName>UNDELIVERABLE</groupName>
            <id>9</id>
            <name>UNDELIVERABLE_NOT_DELIVERED</name>
            <description>Message sent not delivered</description>
         </status>
         <error>
            <groupId>1</groupId>
            <groupName>HANDSET_ERRORS</groupName>
            <id>27</id>
            <name>EC_ABSENT_SUBSCRIBER</name>
            <description>Absent  Subscriber</description>
            <permanent>false</permanent>
         </error>
      </ncResults>
   </ncResults>
</NCResponse>
```

Additional examples

Number Context lookup - Single phone number

Request

JSON

```
POST /restapi/number/1/notify HTTP/1.1
Host: 107.20.199.106
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
Content-Type: application/json
Accept: application/json

{
   "to":[
      "41793026727"
   ],
   "notifyUrl":"http://example.com/notifyUrl",
   "notifyContentType":"application/json"
}
```

XML

```
POST /restapi/number/1/notify HTTP/1.1
Host: 107.20.199.106
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
Content-Type: application/xml
Accept: application/xml

<request>
   <to>41793026727</to>
   <notifyUrl>http://example.com/notifyUrl</notifyUrl>
   <notifyContentType>application/xml</notifyContentType>
</request>
```

cURL

```
curl -X POST \
-H "Content-Type: application/json" \
-H 'Accept: application/json' \
-H "Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ=" \
-d '{
   "to":[
```

```
        "41793026727"
    ],
    "notifyUrl":"http://example.com/notifyUrl",
    "notifyContentType":"application/json"
}'  http://107.20.199.106/restapi/number/1/notify
```

PHP

```
<?php

$request = new HttpRequest();
$request->setUrl('http://107.20.199.106/restapi/number/1/notify');
$request->setMethod(HTTP_METH_POST);

$request->setHeaders(array(
  'content-type' => 'application/json',
  'accept' => 'application/json',
  'authorization' => 'Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ=='
));

$request->setBody('{
    "to":[
        "41793026727"
    ],
    "notifyUrl":"http://example.com/notifyUrl",
    "notifyContentType":"application/json"
}');

try {
  $response = $request->send();

  echo $response->getBody();
} catch (HttpException $ex) {
  echo $ex;
}
```

Ruby

```
require 'uri'
require 'net/http'

url = URI("http://107.20.199.106/restapi/number/1/notify")
```

```
http = Net::HTTP.new(url.host, url.port)
http.use_ssl = true
http.verify_mode = OpenSSL::SSL::VERIFY_NONE

request =Net::HTTP::Post.new(url)
request["authorization"] = 'Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ=='
request["accept"] ='application/json'
request["content-type"] = 'application/json'

request.body = "{\"to\":[\"41793026727\"],
\"notifyUrl\":\"http://example.com/notifyUrl\",
\"notifyContentType\":\"application/json\"}"

response =http.request(request)
puts response.read_body
```

Python

```
import http.client

conn = http.client.httpConnection("107.20.199.106")

payload ="{\"to\":[\"41793026727\"],
\"notifyUrl\":\"http://example.com/notifyUrl\",
\"notifyContentType\":\"application/json\"}"

headers = {
    'authorization': "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==",
    'accept': "application/json",
    'content-type':  "application/json"
    }

conn.request("POST", "/restapi/number/1/notify", payload, headers)

res = conn.getresponse()
data = res.read()

print(data.decode("utf-8"))
```

Java

```
HttpResponse<String> response =
Unirest.post("http://107.20.199.106/restapi/number/1/notify")
```

```
  .header("authorization", "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==")
  .header("accept",  "application/json")
  .header("content-type",  "application/json")
  .body("{\"to\":[\"41793026727\"],
\"notifyUrl\":\"http://example.com/notifyUrl\",
\"notifyContentType\":\"application/json\"}")
  .asString();
```

C#

```
var client = new
RestClient("http://107.20.199.106/restapi/number/1/notify");

var request = new RestRequest(Method.POST);
request.AddHeader("content-type", "application/json");
request.AddHeader("accept",  "application/json");
request.AddHeader("authorization", "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==");
request.AddParameter("application/json",   "{\"to\":[\"41793026727\"],
\"notifyUrl\":\"http://example.com/notifyUrl\",
\"notifyContentType\":\"application/json\"}", ParameterType.RequestBody);

IRestResponse response = client.Execute(request);
```

JavaScript

```
var data = JSON.stringify({
  "to": [
    "41793026727"
  ],
  "notifyUrl": "http://example.com/notifyUrl",
  "notifyContentType":  "application/json"
});

var xhr = new XMLHttpRequest();
xhr.withCredentials = true;

xhr.addEventListener("readystatechange", function () {
  if (this.readyState === this.DONE) {
    console.log(this.responseText);
  }
});

xhr.open("POST",   "http://107.20.199.106/restapi/number/1/notify");
```

```
xhr.setRequestHeader("authorization", "Basic
QWxhZGRpbjpvcGVuIHNlc2FtZQ==");
xhr.setRequestHeader("accept", "application/json");
xhr.setRequestHeader("content-type",   "application/json");

xhr.send(data);
```

Response

JSON

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

{
   "ncResults":[
      {
         "to":"41793026727",
         "status":{
            "groupId":1,
            "groupName":"PENDING",
            "id":3,
            "name":"PENDING_WAITING_DELIVERY",
            "description":"Message sent, waiting for delivery report"
         },
         "messageId":"2b691c32-1233-4716-a763-4f70cc929eae"
      }
   ]
}
```

XML

```
HTTP/1.1 200 OK
Content-Type: application/xml

<NCResponseAsync>
   <ncResults>
      <ncResults>
         <to>41793026727</to>
         <status>
            <groupId>1</groupId>
            <groupName>PENDING</groupName>
            <id>3</id>
```

```
            <name>PENDING_WAITING_DELIVERY</name>
            <description>Message sent, waiting for delivery
report</description>
         </status>
         <messageId>2b691c32-1233-4716-a763-4f70cc929eae</messageId>
      </ncResults>
   </ncResults>
</NCResponseAsync>
```

Number Context lookup - Multiple phone numbers

Request

JSON

```
POST /restapi/number/1/notify HTTP/1.1
Host: 107.20.199.106
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
Content-Type: application/json
Accept: application/json

{
   "to":[
      "41793026727",
      "12345678",
      "Alphanumeric"
   ],
   "notifyUrl":"http://example.com/notifyUrl",
   "notifyContentType":"application/json"
}
```

XML

```
POST /restapi/number/1/notify HTTP/1.1
Host: 107.20.199.106
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
Content-Type: application/xml
Accept: application/xml

<request>
   <to>
      <to>41793026727</to>
```

```
        <to>12345678</to>
        <to>Alphanumeric</to>
    </to>
    <notifyUrl>http://example.com/notifyUrl</notifyUrl>
    <notifyContentType>application/xml</notifyContentType>
</request>
```

cURL

```
curl -X POST
-H "Content-Type: application/json"
-H 'Accept: application/json'
-H "Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ="
-d '{
    "to":[
        "41793026727",
        "12345678",
        "Alphanumeric"
    ],
    "notifyUrl":"http://example.com/notifyUrl",
    "notifyContentType":"application/json"
}'  http://107.20.199.106/restapi/number/1/notify
```

PHP

```
<?php

$request = new HttpRequest();
$request->setUrl('http://107.20.199.106/restapi/number/1/notify');
$request->setMethod(HTTP_METH_POST);

$request->setHeaders(array(
  'content-type' => 'application/json',
  'accept' => 'application/json',
  'authorization' => 'Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ=='
));

$request->setBody('{
    "to":[
        "41793026727",
        "12345678",
        "Alphanumeric"
    ],
```

```
    "notifyUrl":"http://example.com/notifyUrl",
    "notifyContentType":"application/json"
}');

try {
  $response = $request->send();

  echo $response->getBody();
} catch (HttpException $ex) {
  echo $ex;
}
```

Ruby

```
require 'uri'
require 'net/http'

url = URI("http://107.20.199.106/restapi/number/1/notify")

http = Net::HTTP.new(url.host, url.port)
http.use_ssl = true
http.verify_mode = OpenSSL::SSL::VERIFY_NONE

request = Net::HTTP::Post.new(url)
request["authorization"] = 'Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ=='
request["accept"] = 'application/json'
request["content-type"] = 'application/json'

request.body = "{\"to\":[\"41793026727\", \"12345678\", \"Alphanumeric\"],
\"notifyUrl\":\"http://example.com/notifyUrl\",
\"notifyContentType\":\"application/json\"}"

response = http.request(request)
puts response.read_body
```

Python

```
import http.client

conn = http.client.httpConnection("107.20.199.106")

payload = "{\"to\":[\"41793026727\", \"12345678\", \"Alphanumeric\"],
\"notifyUrl\":\"http://example.com/notifyUrl\",
```

```
\"notifyContentType\":\"application/json\"}"

headers = {
    'authorization': "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==",
    'accept': "application/json",
    'content-type':  "application/json"
    }

conn.request("POST", "/restapi/number/1/notify", payload, headers)

res = conn.getresponse()
data = res.read()

print(data.decode("utf-8"))
```

Java

```
HttpResponse<String> response =
Unirest.post("http://107.20.199.106/restapi/number/1/notify")
  .header("authorization", "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==")
  .header("accept",  "application/json")
  .header("content-type",  "application/json")
  .body("{\"to\":[\"41793026727\",  \"12345678\",  \"Alphanumeric\"],
\"notifyUrl\":\"http://example.com/notifyUrl\",
\"notifyContentType\":\"application/json\"}")
  .asString();
```

C#

```
var client = new
RestClient("http://107.20.199.106/restapi/number/1/notify");

var request = new RestRequest(Method.POST);
request.AddHeader("content-type", "application/json");
request.AddHeader("accept",  "application/json");
request.AddHeader("authorization", "Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==");
request.AddParameter("application/json",   "{\"to\":[\"41793026727\",
\"12345678\",  \"Alphanumeric\"],
\"notifyUrl\":\"http://example.com/notifyUrl\",
\"notifyContentType\":\"application/json\"}", ParameterType.RequestBody);

IRestResponse response = client.Execute(request);
```

JavaScript

```
var data = JSON.stringify({
  "to": [
    "41793026727",
    "12345678",
    "Alphanumeric"
  ],
  "notifyUrl": "http://example.com/notifyUrl",
  "notifyContentType":  "application/json"
});

var xhr = new XMLHttpRequest();
xhr.withCredentials = true;

xhr.addEventListener("readystatechange", function () {
  if (this.readyState === this.DONE) {
    console.log(this.responseText);
  }
});

xhr.open("POST", "http://107.20.199.106/restapi/number/1/notify");
xhr.setRequestHeader("authorization", "Basic
QWxhZGRpbjpvcGVuIHNlc2FtZQ==");
xhr.setRequestHeader("accept", "application/json");
xhr.setRequestHeader("content-type",   "application/json");

xhr.send(data);
```

Response

JSON

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

{
   "bulkId":"f5c4322c-10e7-a41e-5528-34fa0b032134",
   "ncResults":[
      {
         "to":"41793026727",
         "status":{
            "groupId":1,
            "groupName":"PENDING",
```

```
            "id":7,
            "name":"PENDING_ENROUTE",
            "description":"Message sent to next instance"
         },
         "messageId":"c2598a78-ba88-423f-8ac3-5dac4af816cf"
      },
      {
         "to":"12345678",
         "status":{
            "groupId":5,
            "groupName":"REJECTED",
            "id":14,
            "name":"REJECTED_DESTINATION",
            "description":"By Destination",
            "action":"Remove destination from blacklist"
         },
         "messageId":"a70437bc-872c-5462-837a-4ede899af3bf"
      },
      {
         "to":"Alphanumeric",
         "status":{
            "groupId":5,
            "groupName":"REJECTED",
            "id":52,
            "name":"REJECTED_DESTINATION",
            "description":"Invalid destination address.",
            "action":"Check to parameter."
         }
      }
   ]
}
```


XML


```
HTTP/1.1 200 OK
Content-Type: application/xml

<NCResponseAsync>
   <ncResults>
      <ncResults>
         <to>41793026727</to>
         <status>
            <groupId>1</groupId>
            <groupName>Pending</groupName>
```

```
            <id>3</id>
            <name>PENDING_WAITING_DELIVERY</name>
            <description>Message sent, waiting for delivery
report</description>
         </status>
         <messageId>c2598a78-ba88-423f-8ac3-5dac4af816cf</messageId>
      </ncResults>
      <ncResults>
         <to>12345678</to>
         <status>
            <groupId>5</groupId>
            <groupName>REJECTED</groupName>
            <id>14</id>
            <name>REJECTED_DESTINATION</name>
            <description>By  Destination</description>
            <action>Remove destination from blacklist</action>
         </status>
         <messageId>a70437bc-872c-5462-837a-4ede899af3bf</messageId>
      </ncResults>
      <ncResults>
         <to>Alphanumeric</to>
         <status>
            <groupId>5</groupId>
            <groupName>REJECTED</groupName>
            <id>52</id>
            <name>REJECTED_DESTINATION</name>
            <description>Invalid destination address.</description>
            <action>Check toparameter.</action>
         </status>
      </ncResults>
   </ncResults>
</NCResponseAsync>
```