Instructions

Date
2017-07-05

Page
1 (13)

Telia Monikanavaviestintä

SMS Messaging

restsms webservice

Company information
Telia Finland Oyj
Teollisuuskatu 15, 00510 HELSINKI, FI
Registered office: Helsinki
Business ID 1475607-9, VAT No. FI14756079

This is a webservice using relatively simple HTTP GET or HTTP POST requests to send SMS, read status of SMS and retrieve incoming SMS.

In the initial implementation, there are three endpoints: /send, /status and /incoming, with the obvious purposes.

Each of the endpoints supports the same function with GET and POST, but in the GET case, parameters are given in the calling URL (after a ? sign, separated by & signs) , while in the POST case the parameters are given in a json document in the HTTP POST request body. UTF-8 encoding is assumed in all HTTP bodies.

Both the GET and the POST return responses in the HTTP response body as a json document.

The format of the input and output json documents and the input url parameters are described below.

The /send endpoint

Used for sending SMS

GET request example e.g. from web browser

```
https://ws.mkv.telia.fi/restsms/lekabrest/send?U=testuser&P=testpass&T=467012
34567,4681234567&F=LEKAB&M=Hall%E5+d%E4r!&2=TRUE&X=CONV123
```

POST request example, probably from an application

```
https://ws.mkv.telia.fi/restsms/lekabrest/send
```

with the contents of the HTTP body:

```
{"username":"testuser","password":"testpass","from":"LEKAB",
 "to":["46701234567","4681234567"],
 "message":"Hallå där!","twoway":true,"conversation":"CONV123"}
```

Explanation of parameters

POST json key	GET query param	json value (strings quoted)	query param value (strings without quotes)	
`Username`	`U`	String	string	username of the VSO SMS sending account
`Password`	`P`	String	string	password of the VSO SMS sending account
`To`	`T`	json list of string	comma separated strings	recipient phone number list
`From`	`F`	String	string	sender id of the SMS (ignored for two-way sms)
`Message`	`M`	string UTF-8	string URL-encoded ISO-8859-1	Message to send
`Twoway`	`2`	boolean (default false)	T, TRUE, Y or YES	sender id from reply number pool

`Conversation`	`X`	String	string	conversation id not sent but echoed with two-way reply
`Costcenter`	`C`	String	string	grouping in billing
`Flash`	`Z`	boolean (default false)	T, TRUE, Y or YES	send flash SMS
`Validminutes`	`V`	string containing integer	string	validity time of sms (default is 24h)

HTTP response

A successful request will return 200 OK and a json document of the following format. Note that the request for status is successful even if the send status of some or all SMS sendings are failed.

```
{
  "accepted" : [ {
    "to" : "46701234567",
    "id" : "354284289"
  } ],
  "rejected" : [ "4681234567" ]
}
```

Note that in this example, the mobile (cell phone, handy) recipient number was accepted while the land-line recipient number was rejected.

Explanation of response

POST json key	json value (strings quoted)	
`accepted`	json list of json documents	list of acknowledged queued SMS, never empty because no queued SMS will cause a HTTP 400 status
`rejected`	json list of string	list of rejected recipients
`to`	string	recipient phone number
`id`	string	id of the SMS (use this for future references to this SMS, e.g. for status queries)

Company information
Telia Finland Oyj
Teollisuuskatu 15, 00510 HELSINKI, FI
Registered office: Helsinki
Business ID 1475607-9, VAT No. FI14756079

Example Python 3 code for the /send endpoint

```
import json
import requests

sendreq =  {"username" : "testuser", "password": "testpass", "from": "Lekab",
"to":["46700112233","46852012345"], "message":"Tjo flöjt!"}
sendreq_json = json.dumps(sendreq)
url = 'https://ws.mkv.telia.fi/restsms/lekabrest/send'
response = requests.post(url, data=sendreq_json)
sendresp = response.json()
for a in sendresp["accepted"]:
    print("SMS to recipient " + a["to"] + " got message id " + a["id"])
for r in sendresp["rejected"]:
    print("Cannot send to " + r)
```

will output

```
SMS to recipient 46700112233 got message id 6205
Cannot send to 46852012345
```

The /status endpoint

Used to retrieve the status of SMS that were sent earlier.

Reading statuses will by default mark them as read, unless the markasread parameter is explicitly set to false

This endpoint can either retrieve unread statuses of SMS messages sent, or retrieve the statuses for a list of given SMS message ids (independent on whether they were read before or not).

Reading the status of a message will by default mark the status as read, but this can be avoided by setting the **markasread (R)** parameter to FALSE. This default value is TRUE both when reading unread statuses and when reading listed statuses. If markasread is set to FALSE, the same statuses will be retrieved again on the next call.

GET request example e.g. from web browser

```
https://ws.mkv.telia.fi/restsms/lekabrest/status?U=testuser&P=testpass&R=FALSE&I=1088,4140,4118,4243,4412
```

POST request example, probably from an application

```
https://ws.mkv.telia.fi/restsms/lekabrest/status
```

with the contents of the HTTP body:

```
{
   "username" : "testuser",
   "password" : "testpass",
   "markasread" : false,
   "id" : [ "1088", "4140", "4118", "4243", "4412" ]
}
```

Explanation of parameters

POST json key	GET query param	json value (strings quoted)	query param value (strings without quotes)	
`username`	`U`	string	string	username of the VSO SMS sending account

password	P	string	string	password of the VSO SMS sending account
id	I	json list of string	comma separated strings	list of ids of the SMS for which send status is to be retrieved
maxnum	N	integer (default 100)	integer	max number of statuses to retrieve (ignored if ids are listed)
markasread	R	boolean (default true)	F, FALSE, N or NO	flag whether the status should be marked as read (defaults to true)

HTTP response

A successful request will return 200 OK and a json document of the following format

```
{
  "statuses" : [ {
    "to" : "46700123456",
    "from" : "46737494333249",
    "id" : "1088",
    "status" : "DELIVERED",
    "statuscode" : "2",
    "conversation" : "",
    "time" : "1467132305000"
  }, {
    "to" : "46705123456",
    "from" : "Lekab",
    "id" : "4243",
    "status" : "QUEUED",
    "statuscode" : "0",
    "conversation" : "",
    "time" : "1476454236000"
  }, {
    "to" : "46702345678",
    "from" : "46737494333295",
    "id" : "4412",
    "status" : "UNDELIVERABLE",
    "statuscode" : "6",
    "conversation" : "74867653486858240",
    "time" : "1477311457000"
  } ],
```

```
  "notfound" : [ "4140", "4118" ]
}
```

Note that in this example, three of the listed statuses were found, while two were not. If the user is not allowed to read a status for a certain id, because that message belongs to another user, this is treated as not found.

Explanation of response

POST json key	json value (strings quoted)	
`Statuses`	json list of json documents	list of send statuses retrieved
`Notfound`	json list of string	list of message ids for which no status could be retrieved
`To`	string	recipient phone number
`From`	string	sender id (number or alphanumeric)
`Id`	string	id of the SMS
`Status`	string	name of the status state of the message (DELIVERED is good)
`Statuscode`	string containing integer "0" to "13"	integer code of the status state of the message
`Conversation`	string	conversation id given when sending the message
`Time`	string containing long integer	timestamp in milliseconds after the epoch (1970-01-01 00:00:00 Z)

Non-final statuses (a later status update from the mobile carrier is quite likely to change the value)

QUEUED (0), SENT (1)

When the status is updated, the "read" flag is cleared to unread status, and the new status will be retrieved the next time reading. Note that when the recipient's phone is turned off, this is where the status is stuck until the phone is turned on or the message expires.

Successful final status (the recipient's phone has supposedly acknowledged receipt of this message)

DELIVERED (2)

Failing final statuses (the recipient will not get this message)

DELETED (3), EXPIRED (4), REJECTED (5), UNDELIVERABLE (6), ABSENTSUBSCRIBER (8), UNKNOWNSUBSCRIBER (9), INVALIDDESTINATION (10), SUBSCRIBERERROR (11), ERROR (13)

Unclear, probably failing final statuses (probably the mobile carrier has lost or dumped the message, but it may suddenly turn up)

ACCEPTED (7), UNKNOWN (12)

Example Python 3 code for the /status endpoint

```
import json
import requests

statuses = {"username" : "testuser", "password": "testpass", "markasread" :
False, "id" : [ "6202", "6203", "6204" ]}
statuses_json = json.dumps(statuses)
url = 'https://ws.mkv.telia.fi/restsms/lekabrest/status'
response = requests.post(url, data=statuses_json)
statusresp = response.json()
for s in statusresp["statuses"]:
    print("id=" + s["id"] + ", status=" + s["status"])
```

will output

```
id=6202, status=DELIVERED
id=6203, status=UNDELIVERABLE
id=6204, status=QUEUED
```

The /incoming endpoint

Used to retrieve incoming SMS that were either sent to a short or long number rented by the user, or to a number pool number in response to a two-way SMS.

Reading incoming SMS will by default mark them as read, unless the markasread parameter is explicitly set to false

This endpoint can either retrieve unread incoming SMS messages, or retrieve incoming SMS listed in a list of message ids (independent on whether they were read before or not).

Reading the a message will by default mark the message as read, but this can be avoided by setting the **markasread (R)** parameter to FALSE. This default value is TRUE both when reading unread messages and when reading id listed messages. If markasread is set to FALSE, the same messages will be retrieved again on the next call.

GET request example e.g. from web browser

```
https://ws.mkv.telia.fi/restsms/lekabrest/incoming?U=testuser&P=testpass&R=FALSE&N=10&G=Y
```

POST request example, probably from an application

```
https://ws.mkv.telia.fi/restsms/lekabrest/incoming
```

with the contents of the HTTP body:

```
{
   "username" : "testuser",
   "password" : "testpass",
   "markasread" : false,
   "maxnum" : 10,
   "getoriginal" : true
}
```

Explanation of parameters

POST json key	GET query param	json value (strings quoted)	query param value (strings without quotes)	

Username	U	string	string (no quotes)	username of the VSO SMS sending account
Password	P	string	string (no quotes)	password of the VSO SMS sending account
Id	I	json list of string	comma separated strings	list of ids of the SMS for which send status is to be retrieved
Maxnum	N	integer (default 100)	integer	max number of statuses to retrieve (ignored if ids are listed)
Markasread	R	boolean (default true)	F, FALSE, N or NO	flag whether the status should be marked as read (defaults to true)
Getoriginal	G	boolean (default false)	T, TRUE, Y or YES	should original SMS text in a two-way conversation be retrieved?

HTTP response

A successful request (at least one SMS was successfully queued for sending) will return 200 OK and a json document of the following format

```
{
  "incoming" : [ {
    "from" : "46701234567",
    "to" : "54321",
    "id" : "1077",
    "message" : "Please send more info about the club",
    "conversation" : "",
    "resptoid" : "",
    "origmess" : "",
    "time" : "1478538376000"
  }, {
    "from" : "46711223344",
    "to" : "46737494333766",
    "id" : "323",
    "message" : "Yes I would love to",
    "conversation" : "67259314888265728",
    "resptoid" : "3403",
    "origmess" : "Will you join us at the pub after?",
    "time" : "1475497511000"
```

```
  } ],
  "notfound" : [ ]
}
```

Note that in this example where no ids are requested, the list of not found ids will always be empty.

Explanation of response

POST json key	json value (strings quoted)	
`Incoming`	json list of json documents	list of incoming sms messages retrieved
`Notfound`	json list of string	list of message ids for which no incoming message could be retrieved
`From`	string	sender phone number
`To`	string	recipient number (long number or short code)
`Id`	string	id of the incoming SMS
`Message`	string	the incoming message text
`Conversation`	string	conversation id associated with this message (two-way SMS response)
`Resptoid`	string	id of original SMS associated with this message (two-way SMS response)
`Origmess`	string	original SMS text associated with this message (two-way SMS response, retrieved if requested)
`Time`	string containing long integer	timestamp in milliseconds after the epoch (1970-01-01 00:00:00 Z)