Apex SMS API Dev Guide

Methods | **Sending SMS via API**

There are two methods that can be used to send SMS:

- **message_send()** for transactional one at a time messages
- **batch_create()** for sending out bulk SMS campaigns to CSV files

message_send()

This method is for sending a single transactional message. Examples of this could be sending a verification code to a user via SMS or another similar type of notification.

Argument	Type	Description
username	string	Your Apex username
password	string	Your Apex password
to	string	The mobile number that must receive the SMS
text	string	Content of the message
from	string	The from number of the message (source address)
report_mask	int	Delivery report request bitmask (see delivery_report_mask_* variables)
report_url	string	URL to call when delivery status changes. More info.
charset	string	Character set to use (defaults to UTF-8)
data_coding	int	Data coding (see data_coding_*)
message_class	int	Optional. Sets the Message Class in DCS field. Accepts values between 0 and 3, for Message Class 0 to 3, A value of 0 sends the message directly to display, 1 sends to mobile, 2 to SIM and 3 to SIM toolkit.
auto_detect_encoding	int	Auto detect the encoding and send appropriately (useful for sending arabic, hindi, unicode etc messages) 1 = on, 0 = off, defaults off.

Result

In the return result, if successful the details key will contain the message ID.
Example output : *{“status”:1,”message”:”Sent”,”details”:”8bfda1a8-5c12-489f-0107-123000000003”}*

Methods | **Sending SMS via API**

batch_create()

This method is for sending bulk SMS campaigns to .CSV, .TXT, .XLS, .XLSX and any other comma delimited file.

Argument	Type	Description
username	string	Your Apex username
password	string	Your Apex password
name	string	A description / name for this batch
throughput	int	Throughput to deliver this batch at (per second)
file_type	string	File type of the upload (csv, xls or zip accepted)
start_time	string	If the batch must be auto-started at a given time, it must be specified here: eg: 2020-03-04 08:00:00

Including your batch file

Due to the limits of GET parameters, using the batch_create() method requires the batch file to be uploaded as a POST variable. The POST variable that must be used is named ‘data’. If the file is compressed using ZIP compression the POST data will need to be base 64 encoded prior to upload.

Pseudo Code

```
batchCsvData = readFile(myBatchFile.csv);
if(batchFileIsZip) {
        batchCsvData = Base64Encode(batchCsvData)
}
postData = “data=“ + batchCsvData;
```

Result

If the batch is successfully created a batch ID will be returned.

SMS Delivery Reports | **via API**

When sending a message through Apex Visibility, you have the option to track the delivery of these messages. If using HTTP, you need to add two additional fields to your request, as below:

- report_url – The HTTP URL for us to request when the status changes
- report_mask – Which status messages you would like to receive

What is the report_mask?

The report mask is what is known as a bit mask field indicating which status messages you want to receive.

What are the different delivery status codes?

- **1** – Delivered
- **2** – Undelivered
- **4** – Queued at network
- **8** – Sent to network
- **16** – Failed at network

How to calculate the report mask?

To calculate the report mask, you need to pick the status codes you want to receive, and then add them together! And that's your report mask.

For example: I want to receive whether my message was delivered or failed. (1, 2 and 16).
report_mask = 1 + 2 + 16 = 19

So in your next request, send report_mask=19 and we'll send you these status' messages where applicable.

Give me an example:

Ok, so we want to send a message, and receive status reports to our server when the message is delivered, or fails.

Our request parameters will look like this:

Parameter	Example Value
username	myusername
password	mypassword
to	27111256849
from	271112221113
text	Hello there, we are testing delivery reports!
report_mask	19
report_url	http://myserver.com/receive_report.php?myMessageId=12345&status=%d

The resulting URL (once we encode the data) for the request will look like this:

http://rnrconnect.sms-console.com/json?action=message_send&username=myusername&password=mypassword&to=27111256849&from=271112221113&text=Hello+there%2C+we+are+testing+delivery+reports%21&report_mask=19&report_url=http%3A%2F%2Fmyserver.com%2Freceive_report.php%3FmyMessageId%3D12345%26status%3D%25d