mitto

SMS API

Developer Guide

DISCLAIMER

MITTO Customer Support

Refer to the Contacting Support topic.

MITTO Product Information

Phone	+41 41 588 05 49
Email	info@mitto.ch
Web Address	http://www.mitto.ch

MITTO Headquarters

Bahnhofstrasse 21
CH 6300
Zug
Switzerland

Contents

Preface

This guide is intended for software developers, solution architects, integration engineers, and similar, who seek to integrate MITTO's SMS Messaging service into their own platform or system. The content assumes that the intended audience is familiar with the REST protocol and, to some degree, with telephony standards.

Contacting Support

If you run into a problem which you cannot solve with the help of this guide, our Support department is at your disposal at support@mitto.ch.

INTRODUCTION

This section explains what MITTO SMS API is and what its capabilities are.

What Is MITTO SMS API?

The SMS API is a REST API for sending SMS messages using MITTO's SMS messaging service. The service is available to MITTO clients.

Capabilities of MITTO SMS API

With MITTO SMS API you can:

- Send SMS messages. You can send a message to one recipient per API call.
- Ability to send SMS body texts longer than 160 characters.
- Control the character encoding of the text messages you send. The supported encoding schemas are UTF-8 and Unicode.
- Receive callback notifications about sent messages.
- Add metadata to message which then gets returned to you in the callbacks.
- Make test API calls. (Test the operation of the API without actually sending SMSs.)

KEY CONCEPTS

This section provides general usage information about the MITTO SMA API, like endpoints, methods, and response format and also discusses the related concept of callbacks / delivery reports.

Endpoint and Method

Messages are sent by making **POST** requests to this endpoint:

```
https://rest.mittoapi.com/sms
```

Except for the response format (which is passed as a URL query parameter, see Response Format and Content for details), the rest of the message-related information (like recipient's MSISDN, text of the message, etc.) and sending options (like the character set encoding) are passed as body parameters.

The data format must be specified as query parameter of the request URL:

```
POST https://rest.mittoapi.com/sms?format=json
```

The JSON representation of request body schema for the mandatory parameters only is:

```
{
  "from": "string",
  "to": "string",
  "body": "string"
}
```

The JSON representation of full the full request body schema (mandatory + optional parameters) is:

```
{
  "from": "string",
  "to": "string",
  "body": "string",
  "type": "string",
  "reference": "string",
  "udh": "string",
  "validity": "int"
  "pid": "int",
  "test": Boolean,
}
```

For information about the query string parameters, refer to Request Parameters.

For examples of concrete API calls, refer to the Using the API section.

Authentication

Authentication is made on every request. To authenticate, **pass your API key in the header of the request** like this:

```
X-Mitto-API-Key: string
```

You can see the authentication also in the examples in the Using the API section.

In addition to authenticating on every request, **you must also have the IP from which the requests are made whitelisted on MITTO's system**. This is an additional security measure enforced on MITTO's part.

Response Format and Content

The response is returned in JSON data format. It is recommended to enforce the format as a query string parameter (like **format**=*json*) to make sure the correct format is received. The response format can be JSON or XML.

The response contains information about the delivery status of the SMS, a timestamp for the delivery, and a message ID plus some other information if some of the optional request parameters were specified in the call. The typical response payload (when the call was made with the mandatory request parameters only) looks like this:

```
{
    "ids": [
        "715118ef01aa4480bbd67324a0459b0b"
    ],
    "timestamp": "2019-04-13T12:07:57.8625574Z",
    "responseCode": 0,
    "responseText": "SMS sent successfully."
}
```

For information about the meaning of the individual parameters, refer to the Response Parameters topic. You may also want to see the response example in the Making Test API Calls topic.

Delivery Reports

Delivery reports about the sent SMSs are available via callbacks to a dedicated callback URL. The callback URL is individual for each MITTO customer. Callbacks are managed from your MITTO customer client configuration and cannot be controlled through the SMS API. Your client configuration depends on your use case scenarios so, in some configurations, the delivery reports functionality may not be available (though it typically is).

In the API, you can tag a message you sent with some metadata information which is returned to you in the callback for that message. You can use metadata to identify individual messages or a group of messages (like when belonging to a campaign, etc.) or provide any other information needed for business or technical purposes. For details on how to use metadata tags in messages, refer to Adding Metadata to the Message and Request Parameters (the **reference** parameter).

For concrete examples, refer to Retrieving Delivery Reports.

See also Delivery Report Error Codes.

Long SMSs

Due to the limitations of GSM/PDU standard, the body text of an SMS message cannot be longer than 160 characters (70 characters if the text is Unicode format). MITTO's SMS API overcomes this limitations by splitting larger messages into several smaller messages, each of which conforms to that standards. This operation is automatic and you don't have to do anything when making the API request – the process is triggered by itself when you send the SMS if the text entered in body of the SMS (the **body** parameter of the request) exceeds the limit.

While the recipient receives several SMSs (concatenated together), you make only one API call. In the response for that call, the IDs of all SMS are returned, so it looks like this:


```
{
    "ids": [
        "7d35863fe6f44695a713d54e6ba0aeb5",
        "4e467e7838954dec9cf6beebf150cd03",
        "45bfa71ae1d0460e9ce00e15ccd3d3c3",
        "487e831825464f04afd46bdbe8f90857",
        "2d9f202b2f084601935a971fd566641f",
        "a606dcad229e41f19f0bc6d89b761912",
        "9e5765e99d9746eab206bcedc3bf8d39",
        "3cd0206933b74421bd4d6dd0119332d2"
    ],
    "timestamp": "2019-04-14T16:55:56.2652450Z",
    "responseCode": 0,
    "responseText": "SMS sent successfully."
}
```


For the entire example, refer to Sending a Long SMS.

> **Note:**
>
> *Concatenating the messages additionally reduces the maximum length of the body text to 153 (67 for Unicode).*

Using the API

This section shows you, with concrete examples, how to send SMS messages. It covers the typical use cases starting from sending a standard SMS and then go into most advanced usages like specifying different text encoding, making test API calls, and adding metadata to the message (returned in the callback) which you can further retrieve and process by your application.

Sending an SMS Message

This example shows sending an SMS saying “Hello, World!” with the default character encoding to recipient +49 172 555 1234.

Request

HTTP

URL

```
POST https://rest.mittoapi.com/sms?format=json
```

Header

```
X-Mitto-API-Key: Xdqnjeewea
...
```

Body

```
{
  "from": "MITTO SMS",
  "to": "491725551234",
  "body": "Hello, World!"
}
```

cURL

```
curl -X POST \
  'https://rest.mittoapi.com/sms?format=json' \
  -H 'X-Mitto-API-Key: Xdqnjeewea' \
  -d '{
        "from": "MITTO SMS",
        "to": "491725551234",
        "body": "Hello, World!"
      }'
```

Response


```
{
    "ids": [
        "715118ef01aa4480bbd67324a0459b0b"
    ],
    "timestamp": "2019-04-13T12:07:57.8625574Z",
    "responseCode": 0,
    "responseText": "SMS sent successfully."
}
```

Changing the Default Encoding of the Message

This example shows sending an SMS saying “Hello, World!” in Russian (“Здравствуй, Мир!”) with Unicode character encoding to recipient +49 172 555 1234. The encoding is specified in the (**type**=*Unicode*) query parameter.

Request

HTTP

URL

```
POST https://rest.mittoapi.com/sms?format=json
```

Header

```
X-Mitto-API-Key: Xdqnjeewea
...
```

Body

```
{
  "from": "MITTO SMS",
  "to": "491725551234",
  "body": "Здравствуй, Мир!"
  "type": "Unicode"
}
```

cURL

```
curl -X POST \
  'https://rest.mittoapi.com/sms?format=json' \
  -H 'X-Mitto-API-Key: Xdqnjeewea' \
  -d '{
        "from": "MITTO SMS",
        "to": "491725551234",
        "body": "Здравствуй, Мир!",
        "type": "Unicode"
      }'
```

Sending a Long SMS

This example shows sending an SMS with a text body that exceeds the maximum allowed number of characters (160 for UTF-8, 70 for Unicode, refer to Long SMSs for details). The text is 534 characters long and is split into 8 concatenated messages as evident from the number of the SMS IDs (the **ids** parameter) returned in the response.

Request

HTTP

URL

```
POST https://rest.mittoapi.com/sms?format=json
```

Header

```
X-Mitto-API-Key: Xdqnjeewea
...
```

Body

```
{
  "from": "MITTO SMS",
  "to": "491725551234",
  "body": "Due to the limitations of GSM/PDU standard, the body text of an
           SMS message cannot be longer than 160 characters (70 characters
           if the text is Unicode format). MITTO's SMS API overcomes this
           limitations by splitting larger messages into several smaller
           messages, each of which conforms to that standards. This
           operation is automatic and you don't have to do anything when
           making the API request - the process is triggered automatically
           when the text entered in body of the SMS (the body parameter of
           the request) exceeds the limit.",
  "type": "Unicode"
}
```

cURL

```
curl -X POST \
  'https://rest.mittoapi.com/sms?format=json' \
  -H 'X-Mitto-API-Key: Xdqnjeewea' \
  -d '{
        "from": "MITTO SMS",
        "to": "491725551234",
        "body": "Due to the limitations of GSM/PDU standard, the body text
                 of an SMS message cannot be longer than 160 characters (70
                 characters if the text is Unicode format). MITTO's SMS API
                 overcomes this limitations by splitting larger messages into
                 several smaller messages, each of which conforms to that
                 standards. This operation is automatic and you don't have to
                 do anything when making the API request - the process is
```

```
                triggered automatically when the text entered in body of the
                SMS (the body parameter of the request) exceeds the limit.",
        "type": "Unicode"
      }'
```

Response


```
{
    "ids": [
        "7d35863fe6f44695a713d54e6ba0aeb5",
        "4e467e7838954dec9cf6beebf150cd03",
        "45bfa71ae1d0460e9ce00e15ccd3d3c3",
        "487e831825464f04afd46bdbe8f90857",
        "2d9f202b2f084601935a971fd566641f",
        "a606dcad229e41f19f0bc6d89b761912",
        "9e5765e99d9746eab206bcedc3bf8d39",
        "3cd0206933b74421bd4d6dd0119332d2"
    ],
    "timestamp": "2019-04-14T16:55:56.2652450Z",
    "responseCode": 0,
    "responseText": "SMS sent successfully."
}
```

Adding Metadata to the Message

This example demonstrates how to tag a message to metadata in order to identify the message as belonging to a particular marketing campaign and to a particular A/B split test version of the marketing copy (text). In the request below, the message's text is the B-version of an A/B text split sent within a campaign named "Campaign 2019-05". The campaign and the split are tagged in the **reference** query parameter (**reference**=*Campaign 2019-05, B-split message*). The tag info will be returned in the callbacks for this message.

Request

HTTP

URL

```
POST https://rest.mittoapi.com/sms?format=json
```

Header

```
X-Mitto-API-Key: Xdqnjeewea
...
```

Body

```
{
  "from": "MITTO SMS",
  "to": "491725551234",
  "body": "Enjoy the stunning Spring 2019 visuals in our new gallery!",
  "reference": "Campaign 2019-05, B-split message"
}
```

cURL

```
curl -X POST \

          'https://rest.mittoapi.com/sms?format=json&from=MITTO%20SMS&to=49
          1725551234&body=%D0%95njoy%20the%20stunning%20Spring%202019%20vis
          uals%20in%20our%20new%20gallery%21&reference=Campaign%202019-05,%
          20B-split%20message' \
  -H 'X-Mitto-API-Key: Xdqnjeewea' \
  -d '{
       "from": "MITTO SMS",
       "to": "491725551234",
       "body": "Enjoy the stunning Spring 2019 visuals in our new
                gallery!",
       "reference": "Campaign 2019-05, B-split message"
      }'
```

Making Test API Calls

To make a test API call (testing the APIs functionality without actually dispatching a message through MITTOs network and delivering to the recipient), set the **test** query parameter to *true* (**test**=*true*). Note that even though no real message delivery is attempted, the **to** parameter (a recipient's number) must be provided and there must be a number value for it.

In the response body of a test call, the test is indicated as **test**=*true* and the **ids** parameter is not returned.

Request

HTTP

URL

```
POST https://rest.mittoapi.com/sms?format=json
```

Header

```
X-Mitto-API-Key: Xdqnjeewea
...
```

Body

```
{
  "from": "MITTO SMS",
  "to": "491725551234",
  "body": "Hello, World!",
  "test": true
}
```

cURL

```
curl -X POST \
  'https://rest.mittoapi.com/sms?format=json' \
  -H 'X-Mitto-API-Key: Xdqnjeewea' \
  -d '{
        "from": "MITTO SMS",
        "to": "491725551234",
        "body": "Hello, World!"
        "test": true
      }'
```

Response

```
{
    "timestamp": "2019-04-13T17:51:06.3091182Z",
    "responseCode": 0,
    "responseText": "SMS sent successfully.",
    "test": true
}
```

Retrieving Delivery Reports

Following are some examples of how to make delivery report requests with the Callback API. The string template for the API supports angle brackets (**<** and **>**), as well as the dollar sign (**$**) as delimiters.

For details about the parameters, refer to Callback Query Parameters.

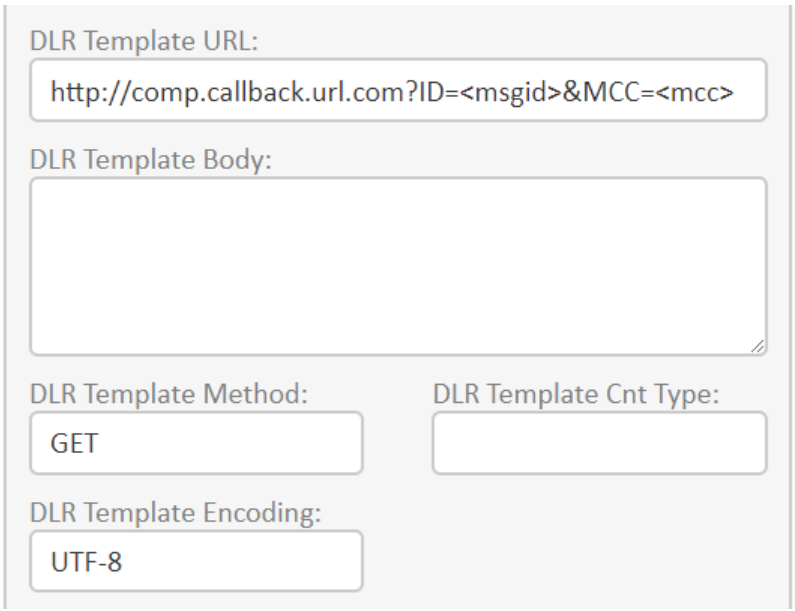
Get Example

Request Model

URL

http://comp.callback.url.com?**ID**=<*msgid*>&**MCC**=<*mcc*>

Callback Interface



Sample Content

http://comp.callback.url.com?ID=d8d9505a-161b-4776-9efb-4939d5b02e99&MCC=262&MNC=07

Request Model

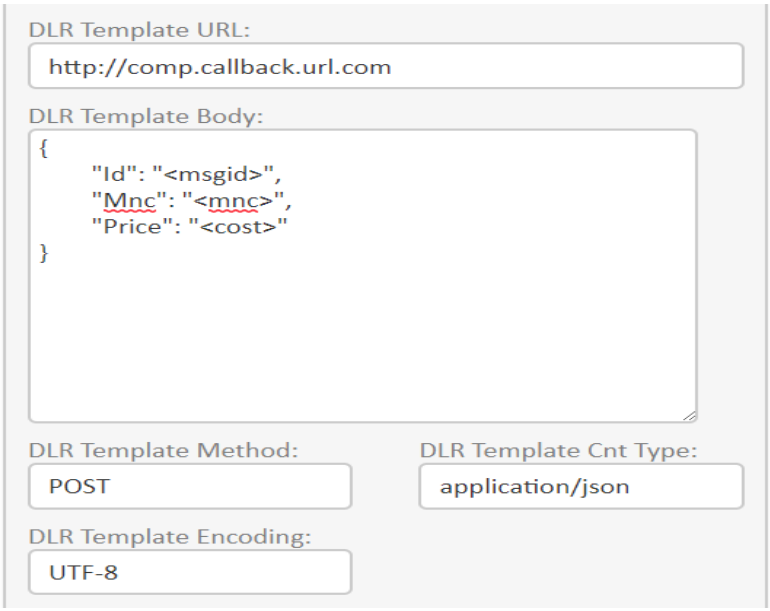
URL

http://comp.callback.url.com

Request Body

```
{
  "Id": "<msgid>",
  "Mnc": "<mnc>",
  "Price": "<cost>"
}
```

Callback Interface



Sample Content

```
{
  "Id": "d8d9505a-161b-4776-9efb-4939d5b02e99",
  "Mnc": "07",
  "Price": "6.00"
}
```

Request Model

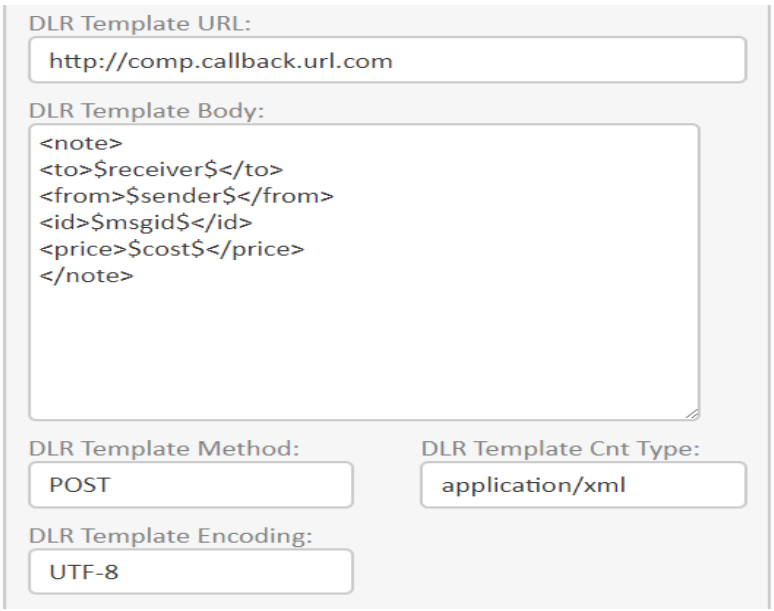
URL

http://comp.callback.url.com

Request Body

```
<note>
  <to>$receiver$</to>
  <from>$sender$</from>
  <id>$msgid$</id>
  <price>$cost$</price>
</note>
```

Callback Interface



Sample Content

```
<note>
  <to>491575800000</to>
  <from>4915758123456</from>
  <id>d8d9505a-161b-4776-9efb-4939d5b02e99</id>
  <price>6.00</price>
</note>
```

Request Model

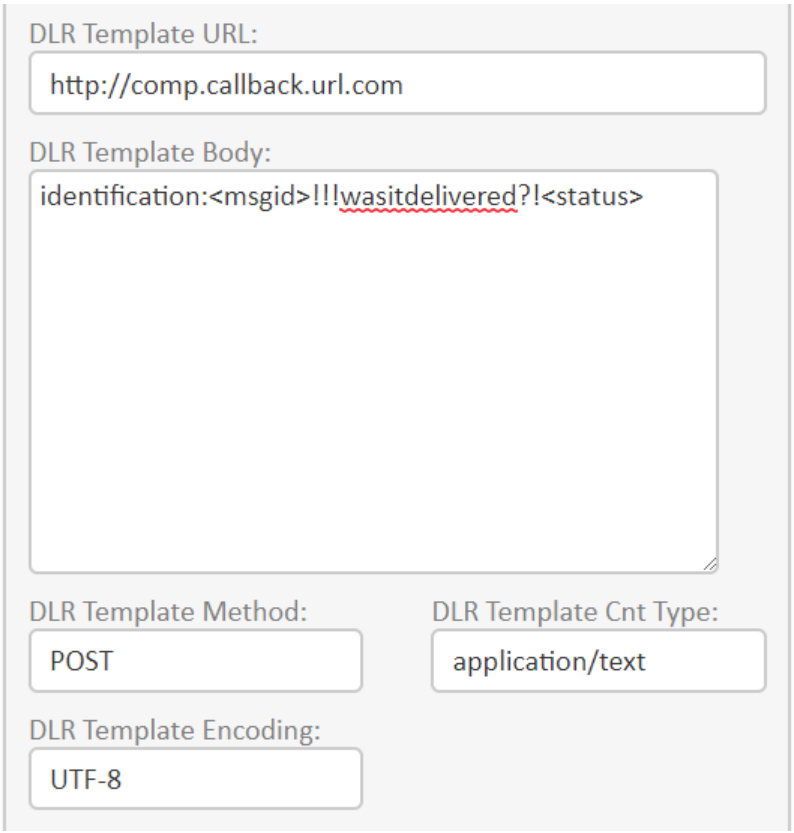
URL

http://comp.callback.url.com

Request Body

```
identification:<msgid>!!!wasitdelivered?!<status>
```

Callback Interface



Sample Content

```
identification: d8d9505a-161b-4776-9efb-4939d5b02e99!!!wasitdelivered?!1
```

API Reference

This section provides detailed information about the request and the response parameters, the request status codes, and the message delivery statuses returned the delivery reports.

Request Parameters

This tables explains the query string parameters of MITTO SMS API.

Parameter Name	Data Type	Mandatory?	Description
api-key	string	yes	Your API key. Set in header of the request.
format	string	yes	Data format of the response. The valid values are *json* and *xml*. Must be passed as a query string parameter.
from	string	yes	Free-form text with which the sender identifies themselves to the recipient. This is what the recipient will see as from whom the message is. Can be a phone number, or the name of the company or the service. Example: *MITTO SMS*
to	string	yes	The number to which the message is sent. Numbers are specified in E.164 format. Example: *359898876737*
body	string	yes	The body of the SMS message. If the message contains characters outside the range of the GSM Standard and Extended tables, then you need to set the character encoding to Unicode (**type**=*Unicode* as a query parameter). For details, refer to Long SMSs and Changing the Default Encoding of the Message.
type	string	no	Character set in which the message body will be encoded. If not specified, the default encoding (UTF-8) is used. For Unicode, set **type**=*Unicode*.
reference	string	no	First metadata field for tagging the message. The metadata is returned in the delivery report by the callback. It can be any free-form text you consider appropriate. You can use a different

			reference string for each message or tag multiple messages with the same string and group them together in this way. For details, refer to Delivery Reports and Adding Metadata to the Message. Example: *Spring campaign*
udh	string	no	Custom Hex-encoded User Data Header. Example: *06050415811581*
validity	int	no	Validity period of the SMS message in minutes. When a message has not been delivered to the receiver at the first attempt, subsequent delivery attempts will be made until the validity period expires, after which the message is discarded as undeliverable. If the specified validity period is different from the validity period of the supplier, then the shorter period is enforced. When not specified, the validity of a message defaults to 2,880 minutes (48 hours).
pid	int	no	Protocol identifier to use. Must be consistent with the **udh** parameter value. When not specified, defaults to *0*.
test	Boolean	no	When set to *true*, the API call is made in test mode (No actual SMS is delivered.) and `"test": true` is returned in the response. For test calls, there is no **ids** parameter in the response. When set to *false*, the API call is made in production mode. An actual SMS is delivered and the **ids** parameter is returned in the response. In addition to that, `"test": true` is also returned. When not specified (the default), the API call is made in production mode. The **ids** parameter is returned in the response, but the **test** parameter is not. See also Making Test API Calls.

Response Parameters

This tables explains the response body parameters of MITTO SMS API.

Parameter Name	Always Returned?	Description
ids	no	ID string of the SMS message. When the length of the body text of the message exceeds the limit (see Long SMSs for details), then several concatenated are send to the receiver and, for this reason, several **ids** parameters are returned in the response. Not returned when making a test call.
timestamp	yes	Timestamp of the SMS message in ISO 8601 format. Example: *2019-04-13T17:51:06.3091182Z* For a description of the format, refer to Date and Time Formats page of the W3 Consortium.
responseCode	yes	A number showing the status of the request. Value of *0* means that the call was successful call, any other value indicates an error. For details, refer to Status Codes.
responseText	yes	Text describing the **responseCode**. For details, refer to Status Codes.
test	no	When `"test": true`, indicates a test API call. (No actual SMS sent.) When not present or when `"test": true`, an actual SMS message has been sent. See also Making Test API Calls.

Status Codes

These are the request status codes and their descriptions returned in the **responseCode** and **responseText** parameters of the response body.

Status Code	Status Code Description	How to fix?
0	Lookup completed successfully.	-
1	Internal error occurred.	Contact MITTO support.
2	Invalid type provided.	Do not use the **type** parameter or use **type**=*Unicode*.
3	Message is empty.	Set the message body text in the **body** parameter of the request. Example: **body**=Hello, World!
4	Message text is invalid.	Make sure the **body** parameter of the request does not contain any characters that are not supported by the encoding set specified in the **type** parameter.
5	Message is too long.	Shorten the message body test to 160 characters or less.
6	Sender is empty.	In the **from** parameter of the request, set some text to identify the sender. Examples: **from**=MITTO%20SMS **from**=John%20Galt
7	Sender is invalid.	Make sure the **from** parameter of the request does not contain any characters that are not supported by the encoding set specified in the **type** parameter.
8	Receiver is empty.	Provide the number to which the message is send as the value of the **to** parameter of the request. Example: **to**=4912312345678
9	Receiver is invalid.	Make sure the value of the **to** parameter of the request contains only numbers and conforms to the E.164 format.

10	Non-supported character set.	Do not use the **type** parameter or use **type**=*Unicode*.
11	Client ID missing.	Contact MITTO support.

Callback Query Parameters

These are the Callback API parameters with which you can retrieve information about a sent SMS. Both GET and POST methods are supported. The string template for the API supports angle brackets (**<** and **>**), as well as the dollar sign (**$**) as delimiters.

Parameter Name	Description
msgid	Message ID
sendtime	Time when message was sent.
status	SMPP delivery status (by spec)
refid	Reference
receiver	Receiver
mcc	Mobile country code (by spec)
mnc	Mobile network code (by spec)
cost	Cost in minor company (customer) currency
errorcode	Error code – see Delivery Report Error Codes.
clientref	Content of the **reference** parameter as specified at sending the SMS.

Message Delivery Statuses

These are the statuses returned in the delivery reports.

States	Description
ACCEPTED	Message has been accepted for delivery.
DELIVERED	Message is delivered to device.
UNDELIVERED	Message is undeliverable. (Maybe device is unavailable for more than 48h or number is wrong.)
BUFFERED	Message is buffered because the device is not available; re-delivery possible in next 48h.
SUBMITTED	Message has been submitted to the service center.
REJECTED	Message is in a rejected state.

Delivery Report Error Codes

These are the error codes used in the message delivery reports. They enable you to more closely monitor delivery outcomes and apply corrective measures when necessary.

Decimal	Status	Description
0	No Error	No Error
1	Unknown subscriber	The MSISDN is inactive or no longer active
2	Unknown subscriber - NR Changed	Fault in Number Portability of the MSISDN of the range holder. Occurs more frequently if number ported multiple times in certain countries
5	Unidentified subscriber	Occurs when the MSC that a message has been sent to is not aware of the subscriber IMSI. Suggests HLR has not been updated or MSC malfunction
6	Absent subscriber for SM	Subscriber handset is currently not available for the SMS service
7	Unknown equipment	Rejection due to subscription handset not identified
8	Roaming not allowed	Rejected due to failed authentication or filtering (roaming scenario)
9	Illegal subscriber	Rejected due to failed authentication or filtering
10	Bearer Service not provisioned	Rejection due to subscription not supporting SMS
11	Teleservice not provisioned	Rejection due to subscription not supporting SMS
12	Illegal equipment	Rejection due to subscription handset or network not supporting SMS
13	Call barred	Rejection due to subscription or network not allowing SMS
21	Facility not supported	Rejection due to subscription not supporting SMS

27	Absent subscriber	Subscriber handset is not logged onto the network due to it being turned off or out of coverage.
28	Incompatible MS terminal error	Rejection due to requested facility not supported by the mobile user terminal
31	Subscriber busy for SM MT	Subscriber handset is currently not available for the SMS delivery.
32	Equipment failure	Receiving handset or equipment does not support SMS or an SMS feature.
33	MS memory capacity exceeded	Receiver handset not having memory capacity to receive the message
34	GSM system failure	MSC - Rejection due to SS7 protocol or network failure
35	GSM Data Error - data missing	GSM data missing
36	GSM Data Error	Wrong GSM data
45	Subscriber busy	Subscriber is currently not able to perform GSM operations.