Zenvia - SMS REST API

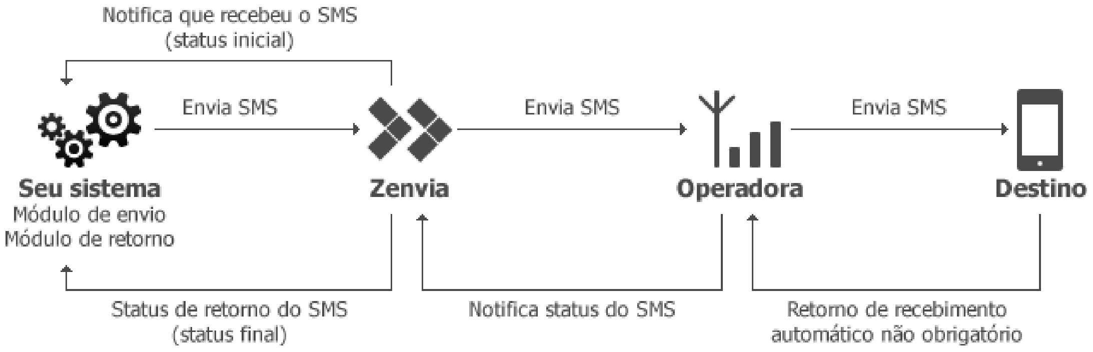
Introduction

Welcome to the Zenvia SMS API. With it you can send SMS to all carriers and receive delivery status and users responses automatically.

For the Portuguese version of this document, please visit Zenvia API SMS!

You can also have access to the most updated version of our API, it does not have all its features yet, but it does allow the use of our multiple channels. For more details visit ZenAPI.

See how simple it is:



The system must make a request to the URL https://api-rest.zenvia.com/services/send-sms, sending the parameters by GET or POST. For each request the response will be a JSON object, which is detailed throughout this documentation.

Firewall e Proxy

If you use firewall or proxy, you must release our IPs list:

Communication	Door

HTTPS	443
FTP	20 and 21
SFTP	22 and 2442

ASN: AS267309

SMS API:

Unset

```
Datacenter Equinix SP2: 45.233.22.224/27
Datacenter Equinix SP3: 45.233.23.224/27
```

DLR's and MO's

Unset

```
Datacenter Equinix/SP2: 45.233.22.225/27
Datacenter Equinix/SP3: 45.233.23.225/27
```

Authentication

Basic Authorization

All the requests to the Zenvia SMS API must have a **Basic Authorization** header. This field should contain your integration user and password, which were sent to you by e-mail. If there is an error during the authentication, the respective error code is returned.

Unset

```
Authorization: Basic Y29udGE6c2VuaGE=
```

The value after the `Basic` is a Base64 key of your user and password. To get the value, use the base64 Linux command:

Unset

```
$ echo -n user:password | base64
Y29udGE6c2VuaGE=
```

The website base64Encode also makes this encoding for free.

Token

To use this authentication method, all requests to the Zenvia SMS API must include the HTTP headers `X-API-Token` and `X-API-Alias`. These fields should contain the access token, which can be generated in the Zenvia platform dashboard, and the integration account we sent via email.

Unset

```
X-API-Token: hKp94crjv9OF3UGrCpSXUJw1-UYHhRvLKNLt
X-API-Alias: your.account
```

Headers

All requests must also include the following headers:

Unset

```
Content-Type:application/json
Accept:application/json
Cache-Control: no-cache
```

Id Parameter

When sending an SMS, we recommend that you use the `id` parameter. It serves as a unique indentifier for your message on our platform and it can be used for status retrieval or even protection against duplicate submissions.

Flash SMS

To send Flash SMS from Zenvia API there is a optional parameter to indicate that the message must be sent as Flash SMS. This parameter is called `flashSms` and accept values `true` (as Flash SMS) or `false`(as normal SMS) and could be informed both in a unique send as in a send to multiple recipients. For compatibility purposes, if the parameter is not informed in the request, the message will be considered a normal SMS.

To use Flash SMS it is necessary to contact our service and request that your account be enabled for this type of send.

Dedicated Short code/LA selection

For accounts that have multiple dedicated short codes, it is possible to select the short code that will be used to send the message through of the optional `sender` parameter. (For more information on the use of dedicated Short code/LA, consult the service/support team)

Limitations

We restricts in 100 simultaneous connections over IP.

Status Table

The API calls will return two types of code: `statusCode` and `detailCode`. The following is the status list supported by our API:

statusCode

Code	Description
00	Ok
01	Scheduled
02	Sent
03	Delivered
04	Not Received

05	Blocked - No Coverage
06	Blocked - Black listed
07	Blocked - Invalid Number
08	Blocked - Content not allowed
08	Blocked - Message Expired
09	Blocked

10	Error

detailCode

Code	Description
000	Message Sent
001	Message Scheduled
002	Message successfully canceled

010	Empty message content
011	Message body invalid
012	Message content overflow
013	Incorrect or incomplete ‘to’ mobile number
014	Empty ‘to’ mobile number
015	Scheduling date invalid or incorrect

016	ID overflow
017	Parameter ‘url’ is invalid or incorrect
018	Field ‘from’ invalid
021	‘id’ fieldismandatory
080	Message with same ID already sent
090	Callback URL is invalid

091	Callback URL overflow
100	Message Queued
110	Message sent to operator
111	Message confirmation unavailable
120	Message received by mobile
130	Message blocked

132	Message already canceled
133	Message content in analysis
134	Message blocked by forbidden content
135	Aggregate is Invalid or Inactive
136	Message expired
140	Mobile number not covered

141	International sending not allowed
145	Inactive mobile number
150	Message expired in operator
160	Operator network error
161	Message rejected by operator
162	Message cancelled or blocked by operator

170	Bad message
171	Bad number
172	Missing parameter
180	Message ID notfound
190	Unknown error
200	Messages Sent

210	Messages scheduled but Account Limit Reached
240	File empty or not sent
241	File too large
242	File readerror
300	Received messages found
301	No received messages found

400	Entity saved
900	Authentication error
901	Account type not support this operation.
990	Account Limit Reached – Please contact support
998	Wrong operation requested
999	Unknown Error

Reference

API Services

Sending a single SMS

Sending a single SMS

This service sends an SMS to the recipient phone. It is possible to send long and short messages.

Short Messages

Not accentuated (up to 160 characters)

Below, an example of the request for short messages:

Unset

```
{
    "sendSmsRequest": {
        "from": "Sender",
        "to": "555199999999",
            "schedule": "2014-08-22T14:55:00",
            "msg": "Test message.",
            "callbackOption": "NONE",
            "callbackUrl":
"https://mycallbackurl.com/callback/zenvia",
            "id": "002",
            "aggregateId": "1111",
            "sender": "30133"
        }
    }
```

In response, the Zenvia API returns a control status:

Unset

```
   {
        "sendSmsResponse" : {
            "statusCode" : "00",
            "statusDescription" : "Ok",
            "detailCode" : "000",
            "detailDescription" : "Message Sent"
        }
    }
```

Another example of synchronous callback:

Unset

```
   {
        "sendSmsResponse" : {
            "statusCode" : "05",
            "statusDescription" : "Blocked",
            "detailCode" : "140",
            "detailDescription" : "Mobile number not covered"
        }
    }
```

Accentuated (up to 70 characters)

It is possible to send short accented messages by adding the parameter `dataCoding`. In this case, short messages are limited to 70 characters and can be sent to the main operators in the market, namely: Vivo, Claro, Tim and Oi.

To learn more about the input parameters, access Single SMS sending.

The characters that are considered special and can only be sent in accented messages, are:

`çÇáéíóúýÁÉÍÓÚÝàèìòùÀÈÌÒÙãõñäëïöüÿÄËÏÖÜÃÕÑâêîôûÂÊÎÔÛºª°“NBSP”`

In addition, it is important to note that double spaces and tabs will always be replaced by a single space.

Below, an example of the request for short accentuated messages:

Unset

```
{
        "sendSmsRequest": {
            "from": "Remetente",
            "to": "555199999999",
            "schedule": "2014-08-22T14:55:00",
            "msg": "Mensagem de teste que terá acento",
            "callbackOption": "NONE",
            "callbackUrl": "https://mycallbackurl.com/callback/zenvia",
            "id": "002",
            "aggregateId": "1111",
            "flashSms": false,
            "dataCoding": 8,
            "sender": "30133"
        }
    }
```

Long messages

Not accentuated (more than 160 characters)

To be able to send long messages it is necessary to contact the support team and request this feature to be enabled in your account.

Every piece of the message contains up to 152 characters, since 8 characters are reserved and used to identify all parts that make up the whole message. This identification is necessary so the devices can interpret all parts as being only one message. Besides that, the partitioning of the messages will be performed in the last space character, i.e., words will not be truncated, and the message can be partitioned with less than 152 characters.

For this type of message, the request is identical to the request to send short messages. Just send the text of the long message in the `msg` attribute with more than 160 characters. Our platform will treat the text of the `msg` attribute as a long message.

There is a limit of 1520 characters when the characters of the attributes `from` and `msg` are added together.

See below an example of a request for long messages:

Unset

```
   {
        "sendSmsRequest": {
            "from": "Remetente",
            "to": "5551999999999",
            "schedule": "2016-12-05T09:18:00",
            "msg": "Lorem ipsum dolor sit amet, consectetur
adipiscing elit. Ut blandit neque consectetur, faucibus tortor
varius, posuere enim. Cras tincidunt lectus eget pulvinar
consectetur. Suspendisse in nibh elit. Ut non pharetra risus, nec
aliquet lectus. Sed convallis mauris vitae lectus consequat, non
mollis nibh hendrerit. Ut tristique commodo ligula eu bibendum.
Quisque vestibulum quis nisi id ultrices.",
            "callbackOption": "NONE",
            "callbackUrl":
"https://mycallbackurl.com/callback/zenvia",
            "id": "idteste",
            "aggregateId": "1111",
            "sender": "30133"
        }
    }
```

In response, the Zenvia API returns a control status. The `parts` object represents all parts of the long message and contains the attributes `partId` and `order`, which represent the identifier and order of each part, respectively:

Unset

```
   {
        "sendSmsResponse" : {
            "statusCode": "00",
            "statusDescription": "Ok",
            "detailCode": "000",
            "detailDescription": "Message Sent",
            "parts": [
                {
                    "partId": "idteste_001",
                    "order": 1
                },
                {
                    "partId": "idteste_002",
                    "order": 2
                },
                {
                    "partId": "idteste_003",
                    "order": 3
                }
            ]
        }
    }
```

Accentuated (more than 70 characters)

It is possible to send long accented messages, adding the parameter `dataCoding`, to the four main operators in the market (Vivo, Claro, Tim and Oi). In this case, the limitation will be 660 characters, adding the characters from the `from` and `msg` attributes. There is a reserve of 4 characters for the identification of the parts that make up the message.

See below an example of a request for long accentuated messages:

Unset

```
{
        "sendSmsRequest": {
            "from": "Remetente",
            "to": "5551999999999",
            "schedule": "2016-12-05T09:18:00",
            "msg": "É possível enviar mensagens longas acentuadas,
adicionando o parâmetro `dataCoding`, para as quatro principais
operadoras do mercado (Vivo, Claro, Tim e Oi). Nesse caso a
limitação será de 660 caracteres somando-se os caracteres dos
atributos `from` e `msg`. Pois há uma reserva de 4 caracteres para a
identificação das partes que compõem mensagem.",
            "callbackOption": "NONE",
            "callbackUrl":
"https://mycallbackurl.com/callback/zenvia",
            "id": "idteste",
            "aggregateId": "1111",
            "flashSms": false,
            "dataCoding": 8,
            "sender": "30133"
        }
    }
```

To receive asynchronous returns, see the API Callbacks menu.

Sending multiple SMS simultaneously

Sending multiple SMS simultaneously

Short messages (up to 70 characters if accented, up to 160 without accent)

This method receives a list of `sendSmsRequest` objects, but in this case the attribute `aggregateId` (if necessary) should be kept out of the individual block, thus being the same for all the messages in the request.

It's recommended to send a maximum of 100 messages per request.

Accented messages without the `dataCoding` parameter, or with this one stating the value `0`, will have their accents removed.

See below an example of a multiple short messages request, the first doesn't contain accent and the second contains:

Unset

```
   {
        "sendSmsMultiRequest":{
            "aggregateId":1750,
            "sendSmsRequestList":
            [
                {
                    "from":"sender",
                    "to":"555199999999",
                    "schedule":"2014-08-23T02:01:23",
                    "msg":"text of message",
                    "callbackOption":"NONE",
                    "callbackUrl":
"https://mycallbackurl.com/callback/zenvia",
                    "id":"003",
                    "sender": "30133"
                },
                {
                    "from":"sender",
                    "to":"555199999999",
                    "schedule":"2014-07-18T02:01:23",
                    "msg":"another text of message",
                    "callbackOption":"NONE",
                    "id":"004",
                    "sender": "21968"
                }
            ]
        }
    }
```

Long non-accented messages (more than 160 characters)

To be able to send long messages it is necessary to contact the support team and request this feature to be enabled in your account.

Every part of the message contains up to 152 characters, since 8 characters are reserved and used to identify all parts that make up the whole message. This identification is necessary so the devices can interpret all parts as being only one message. Besides that, the partitioning of the messages will be performed in the last space character, i.e., words will not be truncated, and the message can be partitioned with less than 152 characters.

For this type of message, the request is identical to the request to send short messages. Just send the text of the long message in the `msg` attribute with more than 160 characters. Our platform will treat the text of the `msg` attribute as a long message.

There is a limit of 1520 characters when the characters of the attributes `from` and `msg` are added together.

This method receives a list of sendSmsRequest objects, but in this case the attribute aggregateId (if necessary) should be kept out of the individual block, thus being the same for all the messages in the request.

Unset

```
   {
        "sendSmsMultiRequest": {
            "aggregateId": "14601",
            "sendSmsRequestList":
            [
                {
                    "from": "Remetente",
                    "to": "5551999999999",
                    "schedule": "2016-12-02T14:42",
                    "msg": "Lorem ipsum dolor sit amet, consectetur
adipiscing elit. Ut blandit neque consectetur, faucibus tortor
varius, posuere enim. Cras tincidunt lectus eget pulvinar
consectetur. Suspendisse in nibh elit. Ut non pharetra risus, nec
aliquet lectus. Sed convallis mauris vitae lectus consequat, non
mollis nibh hendrerit. Ut tristique commodo ligula eu bibendum.
Quisque vestibulum quis nisi id ultrices.",
                    "callbackOption": "NONE",
                    "callbackUrl":
"https://mycallbackurl.com/callback/zenvia",
                    "id": "idteste1",
                    "sender": "30133"
                },
                {
                    "from": "Remetente",
                    "to": "5551999999999",
                    "schedule": "2016-12-02T14:42",
                    "msg": "Lorem ipsum dolor sit amet, consectetur adipiscing elit.",
                    "callbackOption": "NONE",
                    "id": "idteste2",
                    "sender": "21968"
                }
            ]
        }
    }
```

In response, the Zenvia API returns a control status. The `parts` object represents all parts of the long message and contains the attributes `partId` and `order`, which represent the identifier and order of each part, respectively:

Unset

```
   {
        "sendSmsMultiResponse": {
            "sendSmsResponseList": [
                {
                    "statusCode": "00",
                    "statusDescription": "Ok",
                    "detailCode": "000",
                    "detailDescription": "Message Sent",
                    "parts": [
                        {
                            "partId": "idteste1_001",
                            "order": 1
                        },
                        {
                            "partId": "idteste1_002",
                            "order": 2
                        },
                        {
                            "partId": "idteste1_003",
                            "order": 3
                        }
                    ]
                },
                {
                    "statusCode": "00",
                    "statusDescription": "Ok",
                    "detailCode": "000",
                    "detailDescription": "Message Sent",
                    "parts": [
                        {
                            "partId": "idteste2",
                            "order": 1
                        }
                    ]
                }
            ]
        }
    }
```

Long accented messages (more than 70 characters)

In the case of accented messages, each part of the message can contain up to 66 characters, since 4 of these will be reserved for the identification of the parts that compose the message.

For this type of sending, the request will be identical to the request for sending short messages. Just send the text of the long message in the `msg` attribute with more than 70 characters. Our platform will treat the text of the `msg` attribute as a long message.

In the case of accented messages, the limitation will be 660 characters, adding the characters of the `from` and `msg` attributes.

Accented messages without the `dataCoding` parameter, or with this one stating the value `0`, will have their accents removed.

Below is an example of calling short messages, the first with no accent and the second with:

Unset

```
{
        "sendSmsMultiRequest": {
            "aggregateId": "14601",
            "sendSmsRequestList":
            [
                {
                    "from": "Remetente",
                    "to": "5551999999999",
                    "schedule": "2016-12-02T14:42",
                    "msg": "É possível enviar mensagens longas
acentuadas, adicionando o parâmetro `dataCoding`, para as quatro
principais operadoras do mercado (Vivo, Claro, Tim e Oi). Nesse caso
a limitação será de 660 caracteres somando-se os caracteres dos
atributos `from` e `msg`. Pois há uma reserva de 4 caracteres para a
identificação das partes que compõem mensagem.",
                    "callbackOption": "NONE",
                    "callbackUrl":
"https://mycallbackurl.com/callback/zenvia",
                    "id": "idteste1",
                    "flashSms": false,
                    "dataCoding": 8,
                    "sender": "30133"
                },
                {
                    "from": "Remetente",
                    "to": "5551999999999",
                    "schedule": "2016-12-02T14:42",
                    "msg": "Essa mensagem será curta e terá seus
acentos removidos",
                    "callbackOption": "NONE",
                    "id": "idteste2",
                    "flashSms": false,
                    "dataCoding": 0,
                    "sender": "21968"
                }
            ]
        }
    }
```

Get status of an SMS

Get status of an SMS

Consults the delivery status of a message previously sent using your identifier `id`.

Important: consulting an SMS is available for up to 24 hours after your submission.

The following is an example of return:

Unset

```
   {
      "getSmsStatusResp": {
        "id": "006",
        "received": "2014-08-23T02:01:23",
        "shortcode": 69788,
        "mobileOperatorName": "claro",
        "statusCode": "03",
        "statusDescription": "Delivered",
        "detailCode": "120",
        "detailDescription": "Message received by mobile"
      }
    }
```

List new SMS received

List new SMS received

Returns the list of new SMS received. Each SMS will be available just one time. We suggest you save the SMS status in your database for future examinations.

The following is an example of return:

Unset

```
   {
      "receivedResponse": {
        "statusCode": "00",
        "statusDescription": "Ok",
        "detailCode": "300",
        "detailDescription": "Received messages found",
        "receivedMessages": [
          {
            "id": 23190501,
            "dateReceived": "2014-08-22T14:49:36",
            "mobile": "5511991070316",
            "body": "Pare",
            "shortcode": "30133",
            "mobileOperatorName": "Claro",
            "mtId": "hs863223748"
          }
        ]
      }
    }
```

Get SMS received by period

Get SMS received by period

Returns the SMS list received in a defined period.

Unset

```
   {
      "receivedResponse": {
        "statusCode": "00",
        "statusDescription": "Ok",
        "detailCode": "300",
        "detailDescription": "Received messages found",
        "receivedMessages": [
          {
            "id": 23190501,
            "dateReceived": "2014-08-22T14:49:36",
            "mobile": "5511991070316",
            "body": "Pare",
            "shortcode": "30133",
            "mobileOperatorName": "Claro",
            "mtId": "hs863223748"
          }
        ]
      }
    }
```

Warning: To receive the `mobileOperatorName` you must request our service to enabled this in your account.

Scheduled SMS cancellation

Scheduled SMS cancellation

Cancels a scheduled SMS delivery. To carry out the cancellation, the `id` must have been used at the time of the delivery.

Important: cancelling an SMS can only be done up until its schedule date. After that, the SMS will have been sent to the operator and can not be canceled.

The following is an example of return:

Unset

```
   {
        "cancelSmsResp" : {
            "statusCode" : "09",
            "statusDescription" : "Blocked",
            "detailCode” : "002",
            "detailDescription" : "Message successfully canceled"
        }
    }
```

API Callbacks

The Zenvia SMS API can perform callbacks directly to your system, sending delivery status of sent messages or information about SMS received from your customers or contacts.

Callback Delivery Status

The Zenvia platform sends to your system the status of SMS sent. It can send intermediate or final statuses.

To receive the status notification, a notification URL must be configured on the Zenvia platform or the URL must be supplied in the callbackUrl attribute of the sending request. If the callbackUrl attribute is provided in the send request, the notification URL configured in the account will be disregarded and all status notifications for the message sent will be forwarded to the URL provided in the request.

Warning: Please contact our service desk to set your URL callback.

Below, an example of what you will receive in your system:

Unset

```
   {
      "callbackMtRequest": {
        "status": "03",
        "statusMessage": "Delivered",
        "statusDetail": "120",
        "statusDetailMessage": "Message received by mobile",
        "id": "hs765939216",
        "received": "2014-08-26T12:55:48.593-03:00",
        "mobileOperatorName": "Claro"
      }
    }
```

Callback for SMS Received

The Zenvia platform sends to your system an SMS received from a mobile phone. This feature allows you to interact with your contacts in cases like survey, service desk, service evaluation and others.

To receive an SMS in your system, it is necessary to set a notification URL in our platform.

Warning: Please contact our service desk to set your URL callback if you prefer to register your notification URL in your account.

Below, an example of what you will receive in your system:

Unset

```
   {
      "callbackMoRequest": {
        "id": "20690090",
        "mobile": "555191951711",
        "shortCode": "40001",
        "account": "zenvia.envio",
        "body": "Content of reply SMS",
        "received": "2014-08-26T12:27:08.488-03:00",
        "correlatedMessageSmsId": "hs765939061"
      }
    }
```

Callback Retry

Callback retry configurations follow the toward times (in minutos) after the 1º try: 20, 40, 60, 120, 180, 240, 360, 480, 600. Basicly, it means wether some error happen, next try it'll be in 20 minutes. In case a new error happen, the next try it will be in 40 minutes, and on. Time-out in 5 seconds, and after 72 hours, it does not make relationship with MT and MO;

SDK's

To simplify integration with our platform, we offer libraries in the following languages:

PHP

JAVASCRIPT

JAVA

TPS Zenvia

Segue abaixo, alguns números indicando o TPS (transações por segundo) da API REST.

Content Type	Description
Transsactional Traffic	150 TPS - Messages processed and sent by platform to appropriate Carrier within 15 seconds 90% of time

Bulk (Marketing Campaigns	450 TPS - Up to 130k Messages processed and sent by platform to appropriate Carrier within 5 minutes 90% of time, and within 30 minutes or less 99.9%

zenvia.com

We are leaders in the mobile market in Brazil. For over 21 years, we deliver solutions to extend results and generate more interactivity between companies and millions of users in the country