



## GOMOTEXT SMS Gateway API v2.0

### Introduction

This document explains the API offered by GOMOTEXT for developers and Clients to develop their own application or for integrating SMS capabilities into their existing applications

The following APIs are currently supported

- HTTP/HTTPS
- SOAP Web Service

### Features Supported

- [Sending an individual SMS message](#)
- [Sending SMS messages to a predefined list](#)
- [US Carrier Code Lookup](#)
- [Retrieve Send Message history](#)
- [Receiving incoming SMS messages using HTTP](#)

### Sending an individual SMS

#### Using HTTP/HTTPS

The GOMOTEXT SMS Server allows SMS messages to be sent using HTTP. This page details the format of the HTTP requests which can be made to the GOMOTEXT SMS Server.

#### **Sending an SMS Text Message using HTTP GET**

A request for a page using the structure shown below is all that is needed for you to send an SMS through the GOMOTEXT SMS Server. The endpoints for these HTTP/HTTPS requests are listed below

**<http://api.gomotext.com/SMSSend.php>** for HTTP

**<https://api.gomotext.com/SMSSend.php>** for HTTPS (SSL)

Messages should be sent as a HTTP **GET** or **POST** using the parameters listed below. One request should be sent per message.

HTTP/1.1 is enabled so, if sending multiple packets, the TCP/IP connection should be kept open between requests. If sending message with a high transmission rate is required, then several persistent HTTP/1.1 connections should be used concurrently (perhaps 3 or 4).

#### **Parameters for all messages**



Several parameters are common to all message types, and should be included in the HTTP request regardless of the specific method being invoked. Details of these parameters are as follows:

Name	Description
email	Username of the account to send through
password	Password
keyword	Keyword assigned to your account
smsto	MSISDN of the message recipient (eg 12142224444). No leading "+" is required
network	The carrier of the MSISDN, See the complete Carrier list along with the codes (Appendix A)
smsmsg	The message to be send to the user. Needs to be < 140 characters in case listname parameter is used otherwise it can be a long as 160 characters. Please see Appendix B for more information
listname*	Name of the list as defined in the Campaign Manager to the the number needs to be added

(Optional parameters are indicated by a \*)

The following example shows the common structure of all HTTP requests understood by the GOMOTEXT SMS Server:

`http://api.gomotext.com/SMSSend.php?email=email@domain.com&password=myspassword&smsto=12142224444&...other parameters...`

Along with these parameters, there are a variety of other parameters which are needed for the message body, depending on your preference for the encoding/content of the message.

## Parsing responses to GET requests

In each case the responses to GET requests take into account standard internet three digit reply codes: broadly speaking, **20x** implies success, **40x** implies bad request, and **50x** implies server errors.

When a request is successfully received by the GOMOTEXT SMS Server a HTTP success will be returned to the caller (**200** code) and the HTTP body will contain the result of the lookup request.

In the case of a bad request - eg parameters missing/invalid the server will return the error in the body of the response.

If there is a problem with the GOMOTEXT API Service then a 500 Internal server error will be returned. The customer should wait a few seconds and reattempt their request.

### Parsing the response

On a successful request the following parameters will be returned.

Name	Description
msisdn	The MSISDN that the response relates to

login_status	Provides the login status
send_status	Provides the message transmission status
lookup_status	Indicates whether network lookup feature was used
add_status	Indicates whether the number was added to a list in Campaign Manager
keyword_status	Provides the status of the keyword ie; active vs inactive
number_status	Provides the status of the phone number format validation
format_status	Provides the status of the message format validation

### Sample Request

```
GET /SMSSend.php?email=email@domain.com&password=mypassword&keyword=GOMO
&smsto=12142224444&network=TMOBILEUS&smsmsg=Hello&listname=General
```

### Examples of a response

#### Success:

```
<?xml version="1.0"?>
<response>
  <msisdn>+12142224444</msisdn>
  <result>
    <login_status>LOGIN_SUCCESSFUL</login_status>
    <send_status>MESSAGE_SEND_SUCCESSFULLY</send_status>
    <lookup_status>NETWORK_LOOKUP_DISABLED</lookup_status>
    <add_status>MOBILE NUMBER ALREADY EXISTS</add_status>
  </result>
</response>
```

#### Failure:

```
<?xml version="1.0"?>
<response>
  <msisdn>+12142224444</msisdn>
  <result>
    <login_status>LOGIN_SUCCESSFUL</login_status>
    <send_status>MESSAGE_SEND_FAILED</send_status>
    <lookup_status>NETWORK_LOOKUP_DISABLED</lookup_status>
    <format_status>INVALID_MESSAGE_FORMAT</format_status>
  </result>
</response>
```

## Using SOAP Webservice

The GOMOTEXT SMS Server allows SMS messages to be sent using SOAP Webservice. This page details the format of the SOAP requests which can be made to the GOMOTEXT SMS Server.



## WSDL

<http://api.gomotext.com/GomoTextAPI.php?wsdl>

**Name:** *SMSSend*

**Binding:** *GomoTextAPIBinding*

**Endpoint:** *http://api.gomotext.com/GomoTextAPI.php*

**SoapAction:** *urn:GomoTextAPI#SMSSend*

**Style:** *rpc*

**Input:**

*use: encoded*

*namespace: urn:GomoTextAPI*

*encodingStyle: http://schemas.xmlsoap.org/soap/encoding/*

*message: SMSSendRequest*

**parts:**

*input: tns:sendSMSRequest*

**Output:**

*use: encoded*

*namespace: urn:GomoTextAPI*

*encodingStyle: http://schemas.xmlsoap.org/soap/encoding/*

*message: SMSSendResponse*

**parts:**

*return: tns:sendSMSResponse*

**Namespace:** *urn:GomoTextAPI*

**Transport:** *http://schemas.xmlsoap.org/soap/http*

**Documentation:** *Send a SMS using SOAP API*

## Sample Request

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<SOAP-ENV:Envelope SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:si="http://soapinterop.org/xsd"
  xmlns:tns="urn:GomoTextAPI">
  <SOAP-ENV:Body>
    <tns:SMSSend xmlns:tns="urn:GomoTextAPI">
      <input xsi:type="tns:sendSMSRequest">
        <email xsi:type="xsd:string">email@domain.com</email>
        <password xsi:type="xsd:string">mypassword</password>
        <keyword xsi:type="xsd:string">GOMO</keyword>
        <smsto xsi:type="xsd:string">12142224444</smsto>
        <network xsi:type="xsd:string">TMOBILEUS</network>
        <smsmsg xsi:type="xsd:string">Hello World</smsmsg>
        <listname xsi:type="xsd:string">General</listname>
      </input>
    </tns:SMSSend>
  </SOAP-ENV:Body>
```



</SOAP-ENV:Envelope>

**Examples of a response**

**Success:**

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-
ENC="http://schemas.xmlsoap.org/soap/encoding/" xmlns:tns="urn:GomoTextAPI"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance">
<SOAP-ENV:Body>
  <ns1:SMSSendResponse xmlns:ns1="urn:GomoTextAPI">
    <return xsi:type="tns:sendSMSResponse">
      <login_status xsi:type="xsd:string">LOGIN_SUCCESSFUL</login_status>
      <msisdn xsi:type="xsd:string">+12142224444</msisdn>
      <send_status xsi:type="xsd:string">MESSAGE_SEND_SUCCESSFULLY</send_status>
      <add_status xsi:type="xsd:string">MOBILENO AVAILABLE</add_status>
      <keyword_status xsi:type="xsd:string">ACTIVE_KEYWORD</keyword_status>
      <number_status xsi:type="xsd:string">PHONE_NUMBER_VALIDATED</number_status>
      <format_status xsi:type="xsd:string">MESSAGE_TEXT_VALIDATED</format_status>
    </return>
  </ns1:SMSSendResponse>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

**List of Response Messages**

The return fields will support the following possible values

Name
LOGIN_SUCCESSFUL
INVALID_LOGIN_CREDENTIALS
MESSAGE_SEND_SUCCESSFULLY
MESSAGE_SEND_FAILED
NETWORK_LOOKUP_ENABLED
NETWORK_LOOKUP_DISABLED
ACTIVE_KEYWORD
INACTIVE_KEYWORD
PHONE_NUMBER_VALIDATED
INVALID_PHONE_NUMBER_FORMAT
MESSAGE_TEXT_VALIDATED
INVALID_MESSAGE_FORMAT

INVALID_LIST_NAME
ADDLIST_OPTION_NOT_SELECTED
MOBILENO AVAILABLE
MOBILE NUMBER ALREADY EXISTS

## Sending SMS messages to a list

### Using HTTP/HTTPS

The GOMOTEXT SMS Server allows SMS messages to be sent to a preexisting list using HTTP. This page details the format of the HTTP requests which can be made to the GOMOTEXT SMS Server.

#### Sending an SMS Text Message using HTTP GET

A request for a page using the structure shown below is all that is needed for you to send an SMS through the GOMOTEXT SMS Server. The endpoints for these HTTP/HTTPS requests are listed below

<http://api.gomotext.com/SMSListSend.php> for HTTP

<https://api.gomotext.com/SMSListSend.php> for HTTPS (SSL)

Messages should be sent as a HTTP **GET** or **POST** using the parameters listed below. One request should be sent per message.

HTTP/1.1 is enabled so, if sending multiple packets, the TCP/IP connection should be kept open between requests. If sending message with a high transmission rate is required, then several persistent HTTP/1.1 connections should be used concurrently (perhaps 3 or 4).

#### Parameters for all messages

Several parameters are common to all message types, and should be included in the HTTP request regardless of the specific method being invoked. Details of these parameters are as follows:

Name	Description
email	Username of the account to send through
password	Password
keyword	Keyword assigned to your account
smsmsg	The message to be send to the user. Needs to be < 140 characters
listname	Name of the list as defined in the Campaign Manager

The following example shows the common structure of all HTTP requests understood by the GOMOTEXT SMS Server:



`http://api.gomotext.com/SMSListSend.php?email=email@domain.com&password=mypassword&keyword=GOMO&...other parameters...`

Along with these parameters, there are a variety of other parameters which are needed for the message body, depending on your preference for the encoding/content of the message.

## Parsing responses to GET requests

In each case the responses to GET requests take into account standard internet three digit reply codes: broadly speaking, **20x** implies success, **40x** implies bad request, and **50x** implies server errors.

When a request is successfully received by the GOMOTEXT SMS Server a HTTP success will be returned to the caller (**200** code) and the HTTP body will contain the result of the lookup request.

In the case of a bad request - eg parameters missing/invalid the server will return the error in the body of the response.

If there is a problem with the GOMOTEXT API Service then a 500 Internal server error will be returned. The customer should wait a few seconds and reattempt their request.

### Parsing the response

On a successful request the following parameters will be returned.

Name	Description
login_status	Provides the login status
send_status	Provides the message transmission status
keyword_status	Provides the status of the keyword ie; active vs inactive
format_status	Provides the status of the message format validation

### Sample Request

```
GET /SMSListSend.php?email=email@domain.com&password=mypassword&keyword=GOMO
&smsmsg=Hello&listname=General
```

### Examples of a response

#### Success:

```
<?xml version="1.0"?>
<response>
  <msisdn>+12142224444</msisdn>
  <result>
    <login_status>LOGIN_SUCCESSFUL</login_status>
    <send_status>MESSAGE_SEND_SUCCESSFULLY</send_status>
    <lookup_status>NETWORK_LOOKUP_DISABLED</lookup_status>
```



```
        <add_status>MOBILE NUMBER ALREADY EXISTS</add_status>
    </result>
</response>
```

**Failure:**

```
<?xml version="1.0"?>
<response>
    <msisdn>+12142224444</msisdn>
    <result>
        <login_status>LOGIN_SUCCESSFUL</login_status>
        <send_status>MESSAGE_SEND_FAILED</send_status>
        <lookup_status>NETWORK_LOOKUP_DISABLED</lookup_status>
        <format_status>INVALID_MESSAGE_FORMAT</format_status>
    </result>
</response>
```

## Using SOAP Webservice

The GOMOTEXT SMS Server allows SMS messages to be sent to a preexisting list using SOAP Webservice. This page details the format of the SOAP requests which can be made to the GOMOTEXT SMS Server.

### **WSDL**

<http://api.gomotext.com/GomoTextAPI.php?wsdl>

**Name:** *SMSListSend*

**Binding:** *GomoTextAPIBinding*

**Endpoint:** *http://api.gomotext.com/GomoTextAPI.php*

**SoapAction:** *urn:GomoTextAPI#SMSListSend*

**Style:** *rpc*

**Input:**

*use: encoded*

*namespace: urn:GomoTextAPI*

*encodingStyle: http://schemas.xmlsoap.org/soap/encoding/*

*message: SMSListSendRequest*

**parts:**

*input: tns:sendListSMSRequest*

**Output:**

*use: encoded*

*namespace: urn:GomoTextAPI*

*encodingStyle: http://schemas.xmlsoap.org/soap/encoding/*

*message: SMSListSendResponse*

**parts:**

*return: tns:sendListSMSResponse*

**Namespace:** *urn:GomoTextAPI*



Transport: <http://schemas.xmlsoap.org/soap/http>  
Documentation: Send a LIST SMS using SOAP API

### Sample Request

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<SOAP-ENV:Envelope SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:si="http://soapinterop.org/xsd"
  xmlns:tns="urn:GomoTextAPI">
  <SOAP-ENV:Body>
    <tns:SMSListSend xmlns:tns="urn:GomoTextAPI">
      <input xsi:type="tns:sendListSMSRequest">
        <email xsi:type="xsd:string">email@domain.com</email>
        <password xsi:type="xsd:string">mypassword</password>
        <keyword xsi:type="xsd:string">GOMO</keyword>
        <smsmsg xsi:type="xsd:string">Hello World</smsmsg>
        <listname xsi:type="xsd:string">General</listname>
      </input>
    </tns:SMSListSend>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

### Examples of a response

#### Success:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-
ENC="http://schemas.xmlsoap.org/soap/encoding/" xmlns:tns="urn:GomoTextAPI"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance">
  <SOAP-ENV:Body>
    <ns1:SMSListSendResponse xmlns:ns1="urn:GomoTextAPI">
      <return xsi:type="tns:sendListSMSResponse">
        <keyword xsi:type="xsd:string">GOMO</keyword>
        <listname xsi:type="xsd:string">GENERAL</listname>
        <login_status xsi:type="xsd:string">LOGIN_SUCCESSFUL</login_status>
        <send_status xsi:type="xsd:string">MESSAGE_SEND_SUCCESSFULLY</send_status>
        <keyword_status xsi:type="xsd:string">ACTIVE_KEYWORD</keyword_status>
        <format_status xsi:type="xsd:string">MESSAGE_TEXT_VALIDATED</format_status>
      </return>
    </ns1:SMSListSendResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

## List of Response Messages

The return fields will support the following possible values

Name
LOGIN_SUCCESSFUL
INVALID_LOGIN_CREDENTIALS
MESSAGE_SEND_SUCCESSFULLY
MESSAGE_SEND_FAILED
ACTIVE_KEYWORD
INACTIVE_KEYWORD
MESSAGE_TEXT_VALIDATED
INVALID_MESSAGE_FORMAT

## US Carrier Code Lookup

### Using HTTP/HTTPS

The GOMOTEXT SMS Server allows US Carrier Codes to be looked up. This page details the format of the HTTP requests which can be made to the GOMOTEXT SMS Server.

#### US Carrier Code Lookup using HTTP GET

A request for a page using the structure shown below is all that is needed for you to send an SMS through the GOMOTEXT SMS Server. The endpoints for these HTTP/HTTPS requests are listed below

<http://api.gomotext.com/USCarrierLookup.php> for HTTP

<https://api.gomotext.com/USCarrierLookup.php> for HTTPS (SSL)

Messages should be sent as a HTTP **GET** or **POST** using the parameters listed below. One request should be sent per message.

HTTP/1.1 is enabled so, if sending multiple packets, the TCP/IP connection should be kept open between requests. If sending message with a high transmission rate is required, then several persistent HTTP/1.1 connections should be used concurrently (perhaps 3 or 4).

#### Parameters for all messages

Several parameters are common to all message types, and should be included in the HTTP request regardless of the specific method being invoked. Details of these parameters are as follows:

Name	Description
------	-------------



email	Username of the account to send through
password	Password
keyword	Keyword assigned to your account
smsto	The MSISDN for which the network code is required

The following example shows the common structure of all HTTP requests understood by the GOMOTEXT SMS Server:

`http://api.gomotext.com/USCarrierLookup.php?email=email@domain.com&password=mypassword&keyword=GOMO&...other parameters...`

Along with these parameters, there are a variety of other parameters which are needed for the message body, depending on your preference for the encoding/content of the message.

### Parsing responses to GET requests

In each case the responses to GET requests take into account standard internet three digit reply codes: broadly speaking, **20x** implies success, **40x** implies bad request, and **50x** implies server errors.

When a request is successfully received by the GOMOTEXT SMS Server a HTTP success will be returned to the caller (**200** code) and the HTTP body will contain the result of the lookup request.

In the case of a bad request - eg parameters missing/invalid the server will return the error in the body of the response.

If there is a problem with the GOMOTEXT API Service then a 500 Internal server error will be returned. The customer should wait a few seconds and reattempt their request.

### Parsing the response

On a successful request the following parameters will be returned.

Name	Description
login_status	Provides the login status
number_status	Provides the status of the phone number format validation
keyword_status	Provides the status of the keyword ie; active vs inactive
network	The carrier of the MSISDN, See the complete Carrier list along with the codes (Appendix A)

### Sample Request

`GET /USCarrierLookup.php?email=email@domain.com&password=mypassword&keyword=GOMO  
&smsto=12142224444`



## Examples of a response

### Success:

```
<?xml version="1.0"?>
<response>
  <msisdn>>+12142224444</msisdn>
  <result>
    <login_status> LOGIN_SUCCESSFUL </login_status>
    <keyword_status>ACTIVE_KEYWORD </keyword_status>
    <number_status>PHONE_NUMBER_VALIDATED </number_status>
    <network>TMOBILEUS</network>
  </result>
</response>
```

### Failure:

```
<?xml version="1.0"?>
<response>
  <msisdn>>+12142224444</msisdn>
  <result>
    <login_status> LOGIN_SUCCESSFUL </login_status>
    <keyword_status>ACTIVE_KEYWORD </keyword_status>
    <number_status>PHONE_NUMBER_VALIDATED </number_status>
    <network>FAILURE</network>
  </result>
</response>
```

## Using SOAP Webservice

The GOMOTEXT SMS Server allows US Carrier Codes to be lookup up using SOAP Webservice. This page details the format of the SOAP requests which can be made to the GOMOTEXT SMS Server.

### WSDL

<http://api.gomotext.com/GomoTextAPI.php?wsdl>

**Name:** *USCarrierLookup*

**Binding:** *GomoTextAPIBinding*

**Endpoint:** *http://api.gomotext.com/GomoTextAPI.php*

**SoapAction:** *urn:GomoTextAPI#USCarrierLookup*

**Style:** *rpc*

**Input:**

*use: encoded*

*namespace: urn:GomoTextAPI*

*encodingStyle: http://schemas.xmlsoap.org/soap/encoding/*

*message: USCarrierLookupRequest*



parts:  
input: tns:usCarrierLookupRequest  
Output:  
use: encoded  
namespace: urn:GomoTextAPI  
encodingStyle: http://schemas.xmlsoap.org/soap/encoding/  
message: USCarrierLookupResponse  
parts:  
return: tns:usCarrierLookupResponse  
Namespace: urn:GomoTextAPI  
Transport: http://schemas.xmlsoap.org/soap/http  
Documentation: Lookup a US Network Code using SOAP API

### Sample Request

```
<?xml version="1.0" encoding="ISO-8859-1"?>  
<SOAP-ENV:Envelope SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"  
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"  
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"  
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
  xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"  
  xmlns:si="http://soapinterop.org/xsd"  
  xmlns:tns="urn:GomoTextAPI">  
  <SOAP-ENV:Body>  
    <tns:USCarrierLookup xmlns:tns="urn:GomoTextAPI">  
      <input xsi:type="tns:usCarrierLookupRequest">  
        <email xsi:type="xsd:string">email@domain.com</email>  
        <password xsi:type="xsd:string">mypassword</password>  
        <keyword xsi:type="xsd:string">GOMO</keyword>  
        <smsto xsi:type="xsd:string">12142224444</smsto>  
      </input>  
    </tns:USCarrierLookup>  
  </SOAP-ENV:Body>  
</SOAP-ENV:Envelope>
```

### Examples of a response

#### Success:

```
<?xml version="1.0" encoding="ISO-8859-1"?>  
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-  
ENC="http://schemas.xmlsoap.org/soap/encoding/" xmlns:tns="urn:GomoTextAPI"  
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-  
instance">  
  <SOAP-ENV:Body>  
    <ns1:USCarrierLookupResponse xmlns:ns1="urn:GomoTextAPI">  
      <return xsi:type="tns:usCarrierLookupResponse">  
        <login_status xsi:type="xsd:string">true</login_status>  
        <msisdn xsi:type="xsd:string">12142224444</msisdn>  
      </return>  
    </ns1:USCarrierLookupResponse>  
  </SOAP-ENV:Body>  
</SOAP-ENV:Envelope>
```



```
<network xsi:type="xsd:string">TMOBILEUS</network>
<keyword_status xsi:type="xsd:string">>true</keyword_status>
<number_status xsi:type="xsd:string">>true</number_status>
</return>
</ns1:USCarrierLookupResponse>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

## List of Response Messages

The return fields will support the following possible values

Name
LOGIN_SUCCESSFUL
INVALID_LOGIN_CREDENTIALS
ACTIVE_KEYWORD
INACTIVE_KEYWORD
PHONE_NUMBER_VALIDATED
INVALID_PHONE_NUMBER_FORMAT

## Retrieving sent message history

### Using HTTP/HTTPS

The GOMOTEXT SMS Server allows retrieval of API usage history using HTTP. This page details the format of the HTTP requests which can be made to the GOMOTEXT SMS Server.

#### Retrieving API history using HTTP GET

A request for a page using the structure shown below is all that is needed for you to send an SMS through the GOMOTEXT SMS Server. The endpoints for these HTTP requests are **http://api.gomotext.com/GetAPIHistory.php** for HTTP

Messages should be sent as a HTTP **GET** or **POST** using the parameters listed below. One request should be sent per message.

HTTP/1.1 is enabled so, if sending multiple packets, the TCP/IP connection should be kept open between requests. If sending message with a high transmission rate is required, then several persistent HTTP/1.1 connections should be used concurrently (perhaps 3 or 4).

#### Parameters for all messages

Several parameters are common to all message types, and should be included in the HTTP request regardless of the specific method being invoked. Details of these parameters are as follows:



Name	Description
email	Username of the account to send through
password	Password
keyword	Keyword assigned to your account
startdate	Start date from which you wish to retrieve the API Usage History
enddate	End date till which you wish to retrieve the API Usage History <b>Note:</b> If end date is more than 30 days from the start date, only the results for the first 30 days will be returned

The following example shows the common structure of all HTTP requests understood by the GOMOTEXT SMS Server:

`http://api.gomotext.com/GetAPIHistory.php?email=email@domain.com&password=mypassword&smsto=12142224444&...other parameters...`

Along with these parameters, there are a variety of other parameters which are needed for the message body, depending on your preference for the encoding/content of the message.

## Parsing responses to GET requests

In each case the responses to GET requests take into account standard internet three digit reply codes: broadly speaking, **20x** implies success, **40x** implies bad request, and **50x** implies server errors.

When a request is successfully received by the GOMOTEXT SMS Server a HTTP success will be returned to the caller (**200** code) and the HTTP body will contain the result of the lookup request.

In the case of a bad request - eg parameters missing/invalid the server will return the error in the body of the response.

If there is a problem with the GOMOTEXT API Service then a 500 Internal server error will be returned. The customer should wait a few seconds and reattempt their lookup.

### Parsing the response

On a successful request the following parameters will be returned.

Name	Description
id	Internal id using by the GOMOTEXT Server
send_date	Date when the message was sent
keyword	Keyword for the account
mobile_number	Mobile number to which the message was sent
message	Message that was sent
network_lookup	This field indicates whether the GOMOTEXT Server needed to perform the carrier lookup

	due to unavailable or Unknown carrier ID in the incoming request
status	This field has two possible values 0 = Message sending failed 1 = Message sent successfully

#### Sample Request

GET /GetAPIHistory.php?email= email@domain.com &password= mypassword &keyword=GOMO  
&startdate=2006-11-10&enddate=2006-11-15

#### Examples of a response

##### Success:

```
<?xml version="1.0"?>
<response>
<record>
  <id>14</id>
  <send_date>2006-11-12</send_date>
  <keyword>GOMO</keyword>
  <mobile_number>15073800239</mobile_number>
  <message>sample1</message>
  <network_lookup>>false</network_lookup>
  <status>1</status>
</record>
<record>
  <id>15</id>
  <send_date>2006-11-12</send_date>
  <keyword>GOMO</keyword>
  <mobile_number>15073800239</mobile_number>
  <message>sample2</message>
  <network_lookup>>true</network_lookup>
  <status>0</status>
</record>
</response>
```

##### Failure:

```
<?xml version="1.0"?>
<response>
  <error>INVALID_KEYWORD</error>
</response>
```

### Using SOAP Webservice

The GOMOTEXT SMS Server allows API history to be retrieved using SOAP Webservice. This page details the format of the SOAP requests which can be made to the GOMOTEXT SMS Server.

## WSDL

<http://api.gomotext.com/GomoTextAPI.php?wsdl>

**Name:** *GetAPIHistory*

**Binding:** *GomoTextAPIBinding*

**Endpoint:** *http://api.gomotext.com/GomoTextAPI.php*

**SoapAction:** *urn:GomoTextAPI#GetAPIHistory*

**Style:** *rpc*

**Input:**

*use: encoded*

*namespace: urn:GomoTextAPI*

*encodingStyle: http://schemas.xmlsoap.org/soap/encoding/*

*message: GetAPIHistoryRequest*

**parts:**

*input: tns:getAPIHistoryRequest*

**Output:**

*use: encoded*

*namespace: urn:GomoTextAPI*

*encodingStyle: http://schemas.xmlsoap.org/soap/encoding/*

*message: GetAPIHistoryResponse*

**parts:**

*return: tns:getAPIHistoryResponse*

**Namespace:** *urn:GomoTextAPI*

**Transport:** *http://schemas.xmlsoap.org/soap/http*

**Documentation:** *Retrieve API Usage History using SOAP API*

## Sample Request

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<SOAP-ENV:Envelope SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:si="http://soapinterop.org/xsd"
  xmlns:tns="urn:GomoTextAPI">
  <SOAP-ENV:Body>
    <tns:GetAPIHistory xmlns:tns="urn:GomoTextAPI">
      <input xsi:type="tns:getAPIHistory">
        <email xsi:type="xsd:string">email@domain.com</email>
        <password xsi:type="xsd:string">mypassword</password>
        <keyword xsi:type="xsd:string">GOMO</keyword>
        <startdate xsi:type="xsd:string">2007-11-10</startdate>
        <enddate xsi:type="xsd:string">2007-12-10</enddate>
      </input>
    </tns:GetAPIHistory>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```



## Examples of a response

### Success:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-
ENC="http://schemas.xmlsoap.org/soap/encoding/" xmlns:tns="urn:GomoTextAPI"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance">
<SOAP-ENV:Body>
  <ns1:GetAPIHistoryResponse xmlns:ns1="urn:GomoTextAPI">
    <return xsi:type="tns:getAPIHistoryResponse">
      <result SOAP-ENC:arrayType="unnamed_struct_use_soapval[2]" xsi:type="SOAP-
ENC:Array">
        <item>
          <id xsi:type="xsd:string">75</id>
          <send_date xsi:type="xsd:string">2007-11-12</send_date>
          <keyword xsi:type="xsd:string">GOMO</keyword>
          <mobile_number
xsi:type="xsd:string">12142224444</mobile_number>
          <message xsi:type="xsd:string">Hello</message>
          <network_lookup xsi:type="xsd:string">>false</network_lookup>
          <status xsi:type="xsd:string">0</status>
        </item>
        <item>
          <id xsi:type="xsd:string">76</id>
          <send_date xsi:type="xsd:string">2007-11-12</send_date>
          <keyword xsi:type="xsd:string">GOMO</keyword>
          <mobile_number
xsi:type="xsd:string">12142224444</mobile_number>
          <message xsi:type="xsd:string">Hello</message>
          <network_lookup xsi:type="xsd:string">>true</network_lookup>
          <status xsi:type="xsd:string">1</status>
        </item>
      </result>
    </return>
  </ns1:GetAPIHistoryResponse>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

## List of Response Messages

The return fields will support the following possible values

Name
<a href="#">LOGIN_SUCCESSFUL</a>
<a href="#">INVALID_LOGIN_CREDENTIALS</a>

ACTIVE_KEYWORD
INACTIVE_KEYWORD
NO_MATCHES_FOUND
VALID_START_DATE
INVALID_START_DATE
VALID_END_DATE
INVALID_END_DATE

## Receiving SMS messages using HTTP

The GOMOTEXT SMS Server will push received SMS messages by making an HTTP GET to the provided URL to indicate the receipt of an SMS message.

### Message format

On receipt of an SMS we will send an HTTP **GET** request to your server, using the parameters included in the table below:

Parameter	Description
smsfrom	Originator of the message
smsto	Shortcode or MSISDN of virtual mobile/sim card
timestamp	Format: yyyy-MM-dd HH:mm:ss
network*	The carrier of the MSISDN(if known), See the complete Carrier list along with the codes (Appendix A)
smsmsg	Message body (in ISO-8859-15)
smsudh*	UDH

Note that optional parameters are marked with a \*

[http://www.server.com/receive.php?smsfrom=447931123456&smsmsg=Hello+World&smsto=12144445555&timestamp=2002-04-30+21%3A58%3A00](http://www.server.com/receive.php?smsfrom=447931123456&smsmsg>Hello+World&smsto=12144445555&timestamp=2002-04-30+21%3A58%3A00)

### Response to the server

The customer's server should respond with an HTTP **200** responses with a non-empty body. This response must be given in a timely manner (sub 10 seconds) currently we do not abandon a request (waiting for the HTTP response) until 30 seconds have passed but this might be reduced in future.



## **Security considerations**

To ensure the authenticity of the HTTP requests you should only accept requests from the following GOMOTEXT IP address:

**75.125.133.18**

**75.125.133.19**

**75.125.133.20**

**75.125.133.21**

**75.125.133.22**

Appendix A - Carrier Codes

Country	Network	Codes
USA	Alltel	ALLTELUS
USA	AT&T Wireless	ATTUS
USA	Boost USA	BOOSTUS
USA	Cellular One	CELLONEUS
USA	Nextel Communications	NEXTELUS
USA	Sprint PCS	SPRINTUS
USA	T-Mobile USA	TMOBILEUS
USA	Verizon Wireless	VERIZONUS
USA	Virgin USA	VIRGINUS
USA	US Cellular	USCELLULARUS
USA	Cincinnati Bell	CINBELLUS
USA	Appalachian	APPALACHIANUS
USA	Bluegrass Cellular	BLUEGRASSUS
USA	ECIT	ECITUS
USA	Centennial	CENTENNIALUS
USA	Immix	IMMIXUS
USA	Revol	REVOLUS
USA	Unicel	RCCNETWORKUS
USA	West Central	WCENTRALUS
USA	Cellular South	CELLULARSOUTHUS
USA	Cellcom	CELLCOMUS
USA	nTelos	NTELOSUS
USA	ACS Wireless	ACSUS
USA	GCI Communications	GENERALCOMUS
USA	Illinois Valley Cellular	IVCUS
USA	Inland Cellular	INLANDUS
USA	Nex-Tech Wireless	NEXTECHUS
USA	Thumb Cellular	THUMBUS
USA	United Wireless	UNITEDWIRELESSUS
USA	Unknown*	UNKNOWNUS
INTL	Outside USA*	OUTSIDEUS

(Parameters indicated by a \* will incur additional charges)



## Appendix B – Message Encoding

Some of the characters when used in the message body with cause the message delivery to fail unless these characters are URL encoded correct before making the API call.

Below are some of the common characters that require encoding

<b>Illegal characters</b>	
<	Less than
>	Greater than
&	Ampersand
'	Apostrophe
"	Quotation mark
?	Question mark