End User Manual

For

Sanchar SMS HTTP Inbound API

Version 0.1.0

Document Control

This table shows a record of significant changes in the document.

Version	Date	Description of change
V0.1	12.05.2014	________

Table of Contents

Overview

Sanchar SMS HTTP Inbound API module is intended for users, who wants to use API SMS Gateway for sending single or multiple messages using HTTP API format and XML format. This document is developed as per requirement of version 1.3.7 of Sanchar SMS Gateway.

System Overview

HTTP API

HTTP API supports requirements given below.

- Supports all languages with special characters (English, Gujarati, Arabic, Greek etc)
- Supports concatenated messages (long SMS)
- Supports text and Unicode messages.
- Scheduling SMS
- Dynamic sender identity

URL:

You will call the below URL using POST or GET method. **For**

Sanchar SMS:

http://{enterprise.sancharsms.in}/sendsms.jsp

For Reseller:

http://{reseller domain name}/sendsms.jsp

Compulsory Parameter Description

1. user: Valid User Name provided, when user registers for an account
2. password: Valid Password provided in combination with the User name , when user registers for an account.
3. mobiles: comma separated mobile numbers on which message will be delivered.(Example 98XXXXXXXX). Either of mobiles or group name you can use.
4. sms: msgbody parameter refers to the Message content, which needs to be sent to entered Mobile Number. (Example: Written content will be sent.)

Optional Parameter Description

1. clientsmsid: Enter unique SMS serial no (Example: 311). Maximum 20 characters are allowed to enter as serial number. It will be same for comma separated mobile number or group, which involves mobile numbers. If you do not enter clientsmsid, then application will take random number as clientsmsid.
2. groupname: Group name, which you have entered , while creating a group in an application. If you want to send SMS to particular group contacts, then specify group name which you have created in an application.
3. senderid: Sender Id registered for sending messages. If you do not enter any specific sender id, then it will take default sender id, which is assigned to your account.
4. scheduletime: If message needs to be sent to particular recipient, in future, then Date & Time can be scheduled for particular message. Format for date & time is yyyy-mm-dd hh:mm:ss.
5. isallowduplicatemobile: If you want to allow duplicate mobile number, then enter value as yes. If you do not provide value, then it will not allow duplicate mobile numbers. Do not enter **clientsmsid** parameter along with **isallowduplicatemobile** parameter, application will generate random client SMS id for your SMS. If you enter client SMS id along with **isallowduplicatemobile** parameter, then application will consider duplicate mobile no. as duplicate entry and will not process for duplicate mobile no.
6. ndncchecking: If you want to check DND number validation, then enter value as 1. By default, it is 0.

Example to call HTTP API:

1. **Having single mobile number and client SMS id parameter**

   http://{enterprise.sancharsms.in}/sendsms.jsp?user=username&password=password&mobiles=30XXXXXXXX&sms=msgbody&senderid=sanchr&clientsmsid=324554&scheduletime=2013-02-14 12:12:12

   If message is successfully sent, then it will give you the below response:

```
<smslist>
        <sms>
                <smsclientid>324554</smsclientid>
                <messageid>2347878XXXXX</messageid>
                <mobile-no>+30XXXXXXXX</mobile-no>
        </sms>
</smslist>
```

2. **Having mobile number in request:**

   http://{enterprise.sancharsms.in}/sendsms.jsp?user=username&password=password&mobiles=30XXXXXXXX,234XXXXXXXXXX&sms=msg body&senderid=sanchr&scheduletime=2013-02-14 12:12:12

3. **Having group name in request:**

   http://{enterprise.sancharsms.in}/sendsms.jsp?user=username&password=password&sms=msgbody&groupname=any_xyz&senderid=sanchr&scheduletime=2013-02-14 12:12:12

   group name is:any_xyz and mobile numbers included are 30XXXXXXXX, 234XXXXXXX.

- In case 2 and 3, if message is sent successfully, then it will give response as given below.

```
<smslist>
        <sms>
                <smsclientid>32455433</smsclientid>
                <messageid>30XXXXXXXX</messageid>
                <mobile-no>+30XXXXXXXX</mobile-no>
        </sms>
</smslist>
<smslist>
        <sms>
                <smsclientid>32455423</smsclientid>
                <messageid>234XXXXXXXXXX</messageid>
                <mobile-no>+30XXXXXXXX</mobile-no>
        </sms>
</smslist>
```

- In above response:

    smsclientid: You will get smsclientid value as random number.

XML API

URL:

You will call the below URL using POST method. For

Sanchar SMS:

http://{enterprise.sancharsms.in}/sendsms.jsp

For Reseller:

http://{reseller domain name}/sendsms.jsp

For bulk SMS sending write XML data in format given below and post to URL given.

```
<?xml version='1.0'?>

<smslist>

        <sms>

                <user>userid</user>

                <password>password</password>

                <message>message content</message>

                <mobiles>+30XXXXXXXX</mobiles>

                <senderid>Sanchr</senderid>

                <cdmasenderid>25468668545</cdmasenderid>

                <clientsmsid>123</clientsmsid>

                <accountusagetypeid>1</accountusagetypeid>

                <unicode>1</unicode>

                <isallowduplicatemobile>yes</isallowduplicatemobile>

        </sms>
```

```
<sms>
            <user>userid</user>
            <password>password</password>
            <message>message content</message>
            <mobiles>+30XXXXXXXX</mobiles>
            <senderid>Sanchr</senderid>
            <cdmasenderid>919924924701</cdmasenderid>
            <clientsmsid>124</clientsmsid>
            <accountusagetypeid>1</accountusagetypeid>
            <unicode>1</unicode>
            <isallowduplicatemobile>yes</isallowduplicatemobile>
    </sms>
</smslist>
```

Compulsory Parameter Description

1. user: Valid User Name provided, when user registers for an account
2. password: Valid Password provided in combination with the User name , when user registers for an account.
3. mobiles: comma separated mobile numbers on which message will be delivered.(Example 98XXXXXXXX). Either of mobiles or group name you can use.
4. sms: msgbody parameter refers to the Message content, which needs to be sent to entered Mobile Number. (Example: Written content will be sent.)

Optional Parameter Description

1. senderid: sender id. If you do not pass sender id or pass as a blank, it will take default sender id which will assign in your account.
2. cdmasenderid: If you want to send sender id on CDMA mobile.
3. clientsmsid: This is unique number. If you enter for single number in request, you will get the same number in response. If you do not enter, you will get random number in response. Maximum characters length for **clientsmsid** parameter is 20. For comma separated multiple mobile numbers or group name you will get the same number in response.
4. accountusagetypeid: If your account type is service (transactional) SMS then pass 1 and if your SMS account type is promotional, then pass 0.
5. unicode: If you want to send Unicode SMS then pass value 1. If you do not provide it will take 0 mean non-Unicode.
6. isallowduplicatemobile: if you want to allow duplicate mobile number, then enter value as yes. If you do not provide value, then it will not allow duplicate mobile numbers. Do not enter **clientsmsid** parameter along with **isallowduplicatemobile** parameter, application will generate random client SMS id for your SMS. If you enter client SMS id along with **isallowduplicatemobile** parameter, then application will consider duplicate mobile no. as duplicate entry and will not process for duplicate mobile no.
7. ndncchecking: If you want to check DND number validation, then enter value as 1. By default, it is 0.

You will receive below XML response if Message is Successfully Sent.

```
<?xml version="1.0" encoding="iso-8859-1" ?>
        <smslist>
                <sms>
                        <smsclientid>123</smsclientid>
                        <messageid>2001318955</messageid>
                        <mobile-no>+30XXXXXXXX</mobile-no>
                </sms>
        </smslist>
```

You will receive below XML response if Message is not successfully sent and if there is any error.

```
<?xml version="1.0" encoding="iso-8859-1" ?>
<smslist>
        <error>
                <smsclientid>0</smsclientid>
                <error-code>-10018</error-code>
                <error-description>Duplicate Mobile</error-description>
                <mobile-no>+30XXXXXXXX</mobile-no>
                <error-action>1</error-action>
        </error>
</smslist>
```

Error Code:

Error Code	Description
-10002	Invalid User name Or Password
-10017	Invalid Schedule Date
-10019	Inactive User
-10018	Duplicate Mobile No.
-10016	Wrong Message Content.
-10015	Message Not Allow Blank.
-10003	Invalid Mobile No.
-10005	Invalid Senderid
-10011	Version not supported.
-10004	Unknown exception.
-10006	Enter mobile no or Select group
-10007	Invalid Group
-10021	User Name Require
-10022	Password Require
-10026	Client SMS ID Max Limit Exceed.

Sample Code:

1. **PHP**

```
<?php

$xml_data='<xmlversion="1.0"><smslist><sms><user>username</user><password>1
12131</password><message>message
content</message><mobiles>9898000000</mobiles><senderid>Sherif</senderid><cd
masenderid>00201009546310</cdmasenderid><accountusagetypeid>1</accountusaget
ypeid></sms></smslist>';

$URL = "http://{domain name}/sendsms.jsp?";

        $ch = curl_init($URL);
        curl_setopt($ch, CURLOPT_SSL_VERIFYHOST, 0);
        curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, 0);
        curl_setopt($ch, CURLOPT_POST, 1);
        curl_setopt($ch, CURLOPT_ENCODING, 'UTF-8');
        curl_setopt($ch,CURLOPT_HTTPHEADER,array('Content-Type:
        application/xml'));
        curl_setopt($ch, CURLOPT_POSTFIELDS, "$xml_data");
        curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
        $output = curl_exec($ch);
        curl_close($ch);
        print_r($output);

?>
```

2. **JAVA**

```
import org.apache.commons.httpclient.HttpClient;

import org.apache.commons.httpclient.methods.EntityEnclosingMethod;

import org.apache.commons.httpclient.methods.PostMethod;


public class testXML

{

public static  String URL = "http://{domain name}/sendsms.jsp?";

public static void main(String[] args) throws InterruptedException

{

 testXML s = new testXML();

String xmlRequest="<smslist><sms><user>username</user><password>12345678

</password><message>TestMessage</message><mobiles>+30XXXXXXXX</mobiles>

<senderid>Aruhat</senderid><clientsmsid>1</clientsmsid></sms></smslist>";

String xmlResponse = s.sendRequest(xmlRequest);

 }

  public String sendRequest(String strXMLRequest)

{

String strResponseBody = "";

try

{

PostMethod post = new PostMethod(URL);

post.setRequestBody(parseStringToIS(strXMLRequest));
if (strXMLRequest.length() < Integer.MAX_VALUE)
{
post.setRequestContentLength((int) strXMLRequest.length());
}
else
 {
    post.setRequestContentLength(EntityEnclosingMethod.CONTENT_LENGTH_CHUNKED);
  }
    post.setRequestHeader("Content-type", "application/xml");
    HttpClient httpclient = new HttpClient();
    int result = httpclient.executeMethod(post);
    strResponseBody = post.getResponseBodyAsString();
    System.out.println(strResponseBody);
    post.releaseConnection();
}
 catch (Exception e)
 {
 System.out.println(e);
}
return strResponseBody;
}
public static java.io.InputStream parseStringToIS(String xml)
{
 if (xml == null)
 {
return null;
}
xml = xml.trim();
java.io.InputStream in = null;
try {
in = new java.io.ByteArrayInputStream(xml.getBytes("UTF-8"));
}
catch (Exception ex)
{
System.out.println(e);
}
 return in;
}
}
```