I. General Information

In addition to SMS Matrix web (GUI) based SMS/Voice messaging platform, SMS Matrix also offers the SMS/TTS/Voice Gateway. Matrix SMS Gateway provides an easy way to integrate SMS capabilities with your IT system, be it a simple website or a complex messaging or alerting system. It allows to deliver SMS, TTS and Voice messages to phone numbers around the world. SMS Matrix provides coverage to all countries.

SMS Matrix Gateway is extremely reliable and developer friendly; it handles either simple HTTP/HTTPS Post or XML access, which allows usage of any programming language capable of making HTTP posts. So far, access to Matrix SMS Gateway has been successfully implemented and tested with the following programming languages: PHP, Perl, Java, C#, Python, C.

If in doubt, please refer to Perl examples, as they were most thoroughly tested.

Because our service is focused on the North American market, we can provide really easy to use API, and much better customer support than our competition.

SMS Text messages in the Unicode (utf-8) are supported, as long as wireless carrier and phone receiver support given language. When the Unicode is used, most of Asian languages, including Chinese, are supported.

Text to Speech (TTS) and Voice messaging, however not being a part of the ‘SMS’ functionality, are also covered in this document.

Technical Support for the SMS Gateway is available at this email address: support@smsmatrix.com

II. Basic API Functions

Matrix SMS/TTS/Voice Gateway provides the following API functions, all available via HTTP/HTTPS calls:

- Send SMS Message
- Send TTS Message
- Send Voice Message
- Query Message Status
- Cellular Carrier Lookup
- Query Account Balance
- Query Message Fee
Send SMS Message

HTTP/HTTPS call:
URL: http://www.smsmatrix.com/matrix
METHOD: POST
POST VARIABLES:
- username: Account user name (login) – an email address
- password: Account password
- phone: Phone number, or comma delimited list of phone numbers
- group: Group name (as defined in GUI/Groups)
- txt: Text of the SMS message to be sent
- tts: Optional, TTS fallback: 1 or 0 (default)

Either 'phone' or 'group' variable has to be specified (but not both).
'Phone' variable will also accept email addresses (besides numeric phone numbers) for email delivery.
If 'tts' variable is set to 1, system will automatically check if provided phone number is mobile or land-line, so appropriate message (SMS or TTS) will be sent. Otherwise, sending SMS to land-line numbers will result in failure. This feature only works for USA/CAN numbers.

Return value: For each phone number provided in the request, the following text entry is returned:

```
PIN=NNNNNNNNNN
ID=TTTTTTTTTTTTTT
STATUSCODE=NNN
STATUSTXT=TTTTTTTTTTTTTTTTTTTTTTT
TIMESTAMP=NNNNNNNNNNN
```

Each line ends with `\n` character. There is no `\r` character.
ID represents unique string assigned by the Gateway to this SMS message.
It is needed when the status of the message is queried.
STATUSCODE represents three digit error/success code. See Error Codes at the end of this document. Generally any status code from 0 to 399 means success.
STATUSTXT represents textual error/success description, such as 'OK MESSAGE QUEUED'
TIMESTAMP represents a time – number of seconds since January 1\textsuperscript{st}, 1970.

Send TTS Message

HTTP/HTTPS call:
URL: http://www.smsmatrix.com/matrix_tts
METHOD: POST
POST VARIABLES:
- username: Account user name (login) – an email address
- password: Account password
- phone: Phone number
- callerid: Optional, 'caller id' phone number
- language: Optional, see list of supported languages
- gender: Optional, 'male' or 'female' (default)
- response: Optional, 1 or 0, see next chapter
- txt: Text of the SMS message to be sent
Return value: The following text entry is returned:

```
ID=TTTTTTTTTTTTTT
STATUSCODE=NNN
STATUSTXT=TTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTT
TIMESTAMP=NNNNNNNNNNNN
```

Each line ends with `\n` character. There is no `\r` character.

**ID** represents unique string assigned by the Gateway to this SMS message. It is needed when the status of the message is queried.

**STATUSCODE** represents three digit error/success code. See Error Codes at the end of this document. Generally any status code from 0 to 399 means success.

**STATUSTXT** represents textual error/success description, such as 'OK MESSAGE QUEUED'.

**TIMESTAMP** represents a time – number of seconds since January 1\(^{st}\), 1970.

---

**Send Voice Message**

HTTP/HTTPS call:

```
URL: http://www.smsmatrix.com/matrix_voice
METHOD: POST
Content-Type: form-data
```

**POST VARIABLES:**

- **username:** Account user name (login) – an email address
- **password:** Account password
- **phone:** Phone number (or comma delimited list)
- **callerid:** Optional, ‘caller id’ phone number
- **response:** Optional, 1 or 0, see next chapter
- **voicefile:** wave or mp3 file

Return value: For each phone number provided in the request, the following text entry is returned:

```
ID=TTTTTTTTTTTT
STATUSCODE=NNN
STATUSTXT=TTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTT
TIMESTAMP=NNNNNNNNNNNN
```

Each line ends with `\n` character. There is no `\r` character.

**ID** represents unique string assigned by the Gateway to this Voice message. It is needed when the status of the message is queried.

**STATUSCODE** represents three digit error/success code. See Error Codes at the end of this document. Generally any status code from 0 to 399 means success.

**STATUSTXT** represents textual error/success description, such as ’OK MESSAGE QUEUED’

**RESPONSE** represents recipients response (a digit pressed), see next chapter for details

**TIMESTAMP** represents a time – number of seconds since January 1\(^{st}\), 1970.

---

**Send Voice Message using voice URL**

HTTP/HTTPS call:

```
URL: http://www.smsmatrix.com/matrix_voice
METHOD: POST
```

**POST VARIABLES:**

---
Return value: For each phone number provided in the request, the following text entry is returned:

```
ID=TTTTTTTTTTTTT
STATUSCODE=NNN
STATUSTXT=TTTTTTTTTTTTTTTTTTTTTT
TIMESTAMP=NNNNNNNNNNN
```

Each line ends with '\n' character. There is no '\r' character.

**ID** represents unique string assigned by the Gateway to this Voice message. It is needed when the status of the message is queried.

**STATUSCODE** represents three digit error/success code. See Error Codes at the end of this document. Generally any status code from 0 to 399 means success.

**STATUSTXT** represents textual error/success description, such as 'OK MESSAGE QUEUED'

**RESPONSE** represents recipients response (a digit pressed), see next chapter for details

**TIMESTAMP** represents a time – number of seconds since January 1\textsuperscript{st}, 1970.

---

Send Combined: Voice and TTS Message

HTTP/HTTPS call:

- **URL:** http://www.smsmatrix.com/matrix_vtts
- **METHOD:** POST
- **CONTENT_TYPE:** application/json
- **POST VARIABLES:** none
- **JSON BODY:**
  ```json
  {
    username: Account user name (login) – an email address
    password: Account password
    phone: Phone number (or comma delimited list)
    language: Optional, see list of supported languages
    gender: Optional, 'male' or 'female' (default)
    callerid: Optional, 10 digit 'caller id' phone number
    response: Optional, 1 or 0 (default)
    msg: list of entries (voice URL or Text) – see example p. 10
  }
  ```

Return value: For each phone number provided in the request, the following text entry is returned:

```
ID=TTTTTTTTTTTTT
STATUSCODE=NNN
STATUSTXT=TTTTTTTTTTTTTTTTTTTTTT
TIMESTAMP=NNNNNNNNNNN
```
Query Message Status

HTTP/HTTPS call:
  URL: http://www.smsmatrix.com/matrix_status
  METHOD: POST
  POST VARIABLES:
    username: Account user name (login) – an email address
    password: Account password
    id: Message ID returned by /matrix call

Return value: The following text entry is returned:

ID=TTTTTTTTTTTTT
STATUS=NNN
TIMEZONE=PDT
RESPONSE=N
STATUSTXT=TTTTTTTTTTTTTTTTTTTTTTTTT
TIMESTAMP=NNNNNNNNNNNN

Each line ends with `\n` character. There is no `\r` character.
ID represents unique string assigned by the Gateway to this SMS message.
STATUS represents three digit error/success code. See Error Codes at the end of this document. Generally any status code from 0 to 399 means success.
STATUSTXT represents textual error/success description, such as 'OK MESSAGE QUEUED'
RESPONSE represents recipients response (a digit pressed), see next chapter for details
TIMESTAMP represents a time – number of seconds since January 1st, 1970.

Cellular Carrier Lookup

HTTP/HTTPS call:
  URL: http://www.smsmatrix.com/carrier
  METHOD: POST
  POST VARIABLES:
    username: Account user name (login) – an email address
    password: Account password
    phone: 10 digit North American cell phone number

Return value: The following text entry is returned:

CARRIER=TTTTTTTTTTTTT

Function returns cellular carrier's name, if resolved. If name can not be resolved, strings UNKNOWN or ERROR will be returned, e.g. CARRIER=UNKNOWN
Current list of supported (for lookup) cellular carriers is available at this address:
http://www.smsmatrix.com/?sms-carriers
Query Account Balance

HTTP/HTTPS call:
URL: http://www.smsmatrix.com/balance
METHOD: POST
POST VARIABLES:
  username: Account user name (login) – an email address
  password: Account password

Return value: The following text entry is returned:

BALANCE=NNN

Function returns number of credits in user’s account. If the balance can not be retrieved, string ERROR will be returned, e.g. BALANCE=ERROR

Query Message Fee

HTTP/HTTPS call:
URL: http://www.smsmatrix.com/voice_rate
or URL: http://www.smsmatrix.com/sms_rate
or URL: http://www.smsmatrix.com/tts_rate
METHOD: POST
POST VARIABLES:
  phone: Phone number (e.g. 12506546544 or 48774752834)

Return value: The following text entry is returned:

VOICE_RATE=NNN
or
SMS_RATE=NNN
or
TTS_RATE=NNN

Function returns number of credits needed to send message to given phone number. If this phone number is invalid or not supported, applicable string will be returned, e.g. VOICE_RATE=NOT SUPPORTED

III. Optional parameters for Voice and TTS calls

All functions for sending **voice and text to speech** messages support additional, optional parameters.

- **callerid** – caller’s phone number. This number should include country code, e.g. 16042349876
- **repeat** – if set to 1, the recipient will be offered an option to “Press 1 repeat this message or hung up”. This option is especially useful for TTS messages, in which case recipients may not be able to understand well the words being spoken, and they would like to listen again to the message.
- **response** – if set to 1, system will allow the recipient to press a digit on the phone as a response to the message received. For example, you may want to send a message of this kind: 
  
  "Hello Adam, this is dentist Kamranakis. You have an appointment with me, Tuesday, 9am. Press 1 to confirm your appointment or press 2 to cancel. Press 9 to repeat this message."
  
If recipient presses something (any number on the keypad), our system will capture this response, and
it will indicate it in the message status. Typical message status may look like this: “**Voice call answered, duration 18 seconds. R = 1**”.
The response digit will be also returned by Query Message Status (matrix_status()) call.
Please remember than '9' is reserved for 'repeat this message' function, so it should not be used for any other purpose.
If this option is set to 1, the 'repeat' option (if set) will be ignored.

**language** – optional and valid only for TTS messages, can be set to:

- us - US English
- en - US English
- uk - UK English
- jp - Japanese
- ch - Mandarin
- it - Italian
- es - Spanish
- fr - French
- de - German
- pl - Polish
- fi - Finnish
- arabic - Arabic
- ro - Romanian
- no - Norwegian
- tr - Turkish
- sw - Swedish
- gr - Greek
- pt - Portuguese

Case sensitive.
If not specified, the default language will be set to 'us'.

- **gender** – optional and valid only for TTS messages, can be set to ‘male’ or ‘female’. If not specified, the gender will be set to ‘female’. Case sensitive.

At present time, all Gateway calls do work via HTTP Post and Get, however only Post will be supported in the future.

However the parameter name for phone number, in all Gateway calls, is set to 'phone', for the compatibility with previous versions of the API, name ‘pin’ can be used instead.

## IV. Two Way messaging for Voice/TTS – Capturing Recipient's Response

It is possible to send voice or text-to-speech message, that would ask the recipient to make a choice by pressing a number on the dial.

- For instance, a dentist would like to send a message - **appointment reminder**:

  "Hello, this is dentist Adam Smith. Press 1 to confirm your appointment tomorrow at 9am, or press 2 to cancel the appointment. Press 9 to repeat this message."

- Or a car serviceman wants to ask if his customers are happy:

  "Hello, this is your Toyota Service in Cleveland. You visited us last week. We would like to know if were happy with the service we provided."
Press 1 if you were happy, or press 2 if you were not. Press 9 to repeat this message.

If 'response' field is set to 1, our system will deliver voice or TTS message, and it will wait and capture recipient's response. If 'response' field is not set, no response will be expected nor captured.

The recipient would listen to the message, and then press 1 or 2. The number pressed will later show up in the message status, so it will be easy for the sender to see the actual recipient's response.

'9' is always reserved for the 'repeat this message', it should not be used for other purposes.

When sender checks the status of the message in the 'History', the status of the message will include the information what key (digit) was pressed by the recipient.

Example of the message status: "Voice call answered, duration 18 seconds. R = 1"

It means that recipient pressed '1' in a response to the message (e.g. appointment was confirmed).

V. Examples in Perl

Send SMS Message Perl Example

```perl
use LWP::UserAgent;
use HTTP::Request::Common;

my $ua = LWP::UserAgent->new();
my $res = $ua->request
(  
   POST 'http://www.smsmatrix.com/matrix',
   Content_Type => 'application/x-www-form-urlencoded',
   Content      => [ 'username' => 'user66@yahoo.com',
                    'password' => 'pass7878',
                    'phone'   => '12506063167,12508763211',
                    'txt'     => 'This is a test, pls ignore' ]
);
if (!$res->is_error) { die "HTTP Error\n"; }
print "Matrix API Response: " . $res->content . "\n";
```

Send TTS Message Perl Example

```perl
use LWP::UserAgent;
use HTTP::Request::Common;

my $ua = LWP::UserAgent->new();
my $res = $ua->request
(  
   POST 'http://www.smsmatrix.com/matrix_tts',
   Content_Type => 'application/x-www-form-urlencoded',
   Content      => [ 'username' => 'user@hotmail.com',
                    'password' => 'pass2727',
                    'phone'   => '12506063167',
                    'language' => 'en',
                    'gender'  => 'female',
                    'response' => 0,
                    'txt'     => 'Adam, web server is down' ]
);
if (!$res->is_error) { die "HTTP Error\n"; }
print "Matrix API Response: " . $res->content . "\n";
```
Send Voice Message Perl Example

```perl
use LWP::UserAgent;
use HTTP::Request::Common;

my $ua = new LWP::UserAgent;
my $res = $ua->request
( POST 'http://www.smsmatrix.com/matrix_voice',
  Content_Type => 'form-data',
  Content => { username => 'user@hotmail.com',
               password => 'pass72727',
               phone => '12502795621',
               voicefile => ['/sounds/nov2009/sale2_2.wav'],
               callerid => '6308891723' ## optional }
};

if ($res->is_error) { die "HTTP Error\n"; }
print "Matrix API Response1: " . $res->content . "\n";
```

Send Voice Message with voice URL Perl Example

```perl
use LWP::UserAgent;
use HTTP::Request::Common;

my $ua = new LWP::UserAgent;
my $res = $ua->request
( POST 'http://www.smsmatrix.com/matrix_voice',
  Content_Type => 'application/x-www-form-urlencoded',
  Content => { username => 'user@hotmail.com',
               password => 'pass72727',
               phone => '12502795621',
               voiceurl => 'http://www.myweb.ca/sounds/welcome1.mp3',
               response => 1,
               callerid => '6308891723' ## optional }
};

if ($res->is_error) { die "HTTP Error\n"; }
print "Matrix API Response1: " . $res->content . "\n";
```

Query Message Status Perl Example

```perl
use LWP::UserAgent;
use HTTP::Request::Common;

my $ua = new LWP::UserAgent;
my $res = $ua->request
( POST 'http://www.smsmatrix.com/matrix_status',
  Content_Type => 'application/x-www-form-urlencoded',
  Content => { username => 'user@hotmail.com',
               password => 'pass72727',
               id => 'a876f1267536589be789effa879' }
};

if ($res->is_error) { die "HTTP Error\n"; }
print "Matrix API Response1: " . $res->content . "\n";
```
**Send Combined: Voice and TTS Message Perl example**

```perl
use LWP::UserAgent;
use HTTP::Request::Common;
use JSON();

my $msg = {
    username => 'user55@hotmail.com',
    password => 'mypass',
    phone => '12502012503', # comma delimited list
    language => 'us', # optional, see list of languages
    gender => 'male', # optional
    callerid => '12501112233', # optional
    response => 0, # optional
    msg => ['http://www.myserver.com/voices/welcome_prompt.wav',
            'This text will be converted to voice by TTS engine.',
            'http://www.myserver.com/voices/segment2.mp3']
};

my $ua = new LWP::UserAgent;
my $res = $ua->request(
    POST 'http://www.smsmatrix.com/matrix_vtts',
    Content_Type => 'application/json',
    Content => JSON::to_json($msg)
);

if ($res->is_error) { die "HTTP Error\n"; }
print "Matrix API Response:\n" . $res->content . "\n";
```
**Cellular Carrier Look-up Perl Example**

```perl
use LWP::UserAgent;
use HTTP::Request::Common;

my $ua = LWP::UserAgent->new();
my $res = $ua->request(
    POST 'http://www.smsmatrix.com/carrier',
    Content_Type => 'application/x-www-form-urlencoded',
    Content     => { 'username' => 'user78@yahoo.ca',
                     'password' => 'passui8888',
                     'phone'   => '2504447766' }
); if ($res->is_error) { die "HTTP Error\n"; }
print "Matrix API Response: " . $res->content . "\n";
```

**Query Account Balance**

```perl
use LWP::UserAgent;
use HTTP::Request::Common;

my $ua = LWP::UserAgent->new();
my $res = $ua->request(
    POST 'http://www.smsmatrix.com/balance',
    Content_Type => 'application/x-www-form-urlencoded',
    Content     => { 'username' => 'user77@gmail.com',
                     'password' => 'mypass' }
); if ($res->is_error) { die "HTTP Error\n"; }
print "Matrix API Response: " . $res->content . "\n";
```

**Query Message Rate**

```perl
use LWP::UserAgent;
use HTTP::Request::Common;

my $ua = LWP::UserAgent->new();
my $res = $ua->request(
    POST 'http://www.smsmatrix.com/sms_rate',
    Content_Type => 'application/x-www-form-urlencoded',
    Content     => { 'phone' => '12506053342' }
); if ($res->is_error) { die "HTTP Error\n"; }
print "Matrix API Response: " . $res->content . "\n";
```

**VI. Other Examples**

**Send SMS Message PHP Example**

```php
<?php
$url = 'http://www.smsmatrix.com/matrix';
$pin = '12506063167'; // comma separated list of phone numbers
// $pin = '12506063167,12508763211';
$username = urlencode ('user@hotmail.com');
$password = urlencode ('pass72727');
$txt = urlencode ('This is a test, pls ignore');

$Q = "$url?username=$username&password=$password&pin=$pin&txt=$txt";

$res = implode ('', file ($Q));
echo "Matrix API Response :\n$res\n";
?>
```
Send SMS Message Java Example

try {
    String MATRIXURL = "http://www.smsmatrix.com/matrix";
    String PIN = "12506063167";
    String USERNAME = "user@hotmail.com";
    String PASSWORD = "pass72727";
    String TXT = "This is a test, pls ignore";

    String q = "username=" + URLEncoder.encode (USERNAME, "UTF-8") + "&" + "password=" + URLEncoder.encode (PASSWORD, "UTF-8") + "&" + "pin=" + PIN + "&" + "txt=" + URLEncoder.encode (TXT, "UTF-8") + ";

    URL url = new URL (MATRIXURL);
    URLConnection conn = url.openConnection();
    conn.setDoOutput (true);
    OutputStreamWriter wr = new OutputStreamWriter (conn.getOutputStream());
    wr.write (q);
    wr.flush();
    BufferedReader rd = new BufferedReader (new InputStreamReader (conn.getInputStream()));
    String line;
    System.out.println ("Matrix API Response :");
    while ((line = rd.readLine()) != null) { System.out.println (line); }
    wr.close();
    rd.close();
} catch (Exception e) { }
Send SMS Message C# Example

```csharp
string MATRIXURL = "http://www.smsmatrix.com/matrix";
string PIN = "12506063167";
string USERNAME = Server.UrlEncode ("user@hotmail.com");
string PASSWORD = Server.UrlEncode ("pass72727");
string TXT = Server.UrlEncode ("This is a test, pls ignore");

string q = "username=" + USERNAME + "&password=" + PASSWORD + "&pin=" + PIN + "&txt=" + TXT;

HttpWebRequest req = (HttpWebRequest)WebRequest.Create (MATRIXURL);
req.Method = "POST";
req.ContentType = "application/x-www-form-urlencoded";
req.ContentLength = q.Length;

StreamWriter streamOut = new StreamWriter(req.GetRequestStream(), System.Text.Encoding.ASCII);
streamOut.Write (q);
streamOut.Close();

StreamReader streamIn = new StreamReader(req.GetResponse().GetResponseStream());
string res = streamIn.ReadToEnd();
Console.WriteLine ("Matrix API Response:

" + res);
streamIn.Close();
```

VII. Error Codes

The following numerical values are returned as STATUS:

<table>
<thead>
<tr>
<th>STATUS</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>OK</td>
</tr>
<tr>
<td>404</td>
<td>ACCOUNT OR USER DOES NOT EXIST</td>
</tr>
<tr>
<td>502</td>
<td>PIN IN DO NOT CALL DATABASE</td>
</tr>
<tr>
<td>503</td>
<td>INSUFFICIENT BALANCE</td>
</tr>
<tr>
<td>504</td>
<td>DATABASE ERROR</td>
</tr>
<tr>
<td>505</td>
<td>USER NOT FOUND OR WRONG PASSWORD</td>
</tr>
<tr>
<td>506</td>
<td>ACCOUNT NOT ACTIVE</td>
</tr>
<tr>
<td>507</td>
<td>DATABASE ERROR</td>
</tr>
<tr>
<td>508</td>
<td>DATABASE ERROR</td>
</tr>
<tr>
<td>510</td>
<td>INVALID USERNAME</td>
</tr>
<tr>
<td>511</td>
<td>INVALID TXT</td>
</tr>
<tr>
<td>512</td>
<td>INVALID PASSWORD</td>
</tr>
<tr>
<td>513</td>
<td>INVALID PIN</td>
</tr>
<tr>
<td>520</td>
<td>ERROR PARSING XML</td>
</tr>
</tbody>
</table>

All values from 0 - 399 (inclusive) mean success, all other values mean failure.
VIII. Incoming SMS (Virtual Mobile Phone Number)

For all incoming SMS messages (when using virtual mobile phone number feature), our system can submit HTTP POST to an URL associated with the number.

The following data (HTTP POST variables) will be submitted:

- phonefrom : phone number of the sender (if available)
- phoneto : virtual phone number
- id : unique alphanumerical ID
- orgserver : our hostname (usually www.smsmatrix.com)
- timestamp : UNIX timestamp (PST)
- txt : body of the SMS

IX. Languages supported by TTS

- us - US English
- en - US English
- uk - UK English
- jp - Japanese
- ch - Mandarin
- it - Italian
- es - Spanish
- fr - French
- de - German
- pl - Polish
- fi - Finnish
- arabic - Arabic
- ro - Romanian
- no - Norwegian
- tr - Turkish
- sw - Swedish
- gr - Greek
- pt - Portuguese

All languages support male/female voice (gender variable in HTTP post)