

# Messagenie Developer's Guide

## Data API Protocol v.1.1

# Contents

1. Audience
2. Basic assumptions
3. Authentication
4. API calls
  - 4.1. List messages in folder
  - 4.2. Show message info
  - 4.3. Set message as read
  - 4.4. Move message to another folder or delete message
  - 4.5. Send message
5. Example code
  - 5.1. PHP
6. Revision history

## 1. Audience

This documentation is intended for programmers who are writing client applications that interact with Messagenie. It provides examples of basic API operations using raw HTTP GET requests.

## 2. Basic assumptions

This documentation assumes that you understand the general principles behind HTTP GET requests and can execute such calls from your application.

Messagenie APIs provide a simple, standard protocol for sending and receiving SMS. All API calls must be requested from **data.php** URL.

Character displayed as `\n` is the new line character (NL, ASCII code: 10).

## 3. Authentication

API call authentication provided via HTTP GET parameter **api\_key**. You must setup API key via standard Messagenie web interface under “**Setting > API Setup**” option. Once API key setup you must include it with every API call from your application.

In the following API structure examples it is assumed that `API_KEY` variable is the API key used for authentication. In the following API examples it is assumed that API key is set to **myapikey**.

## 4. API calls

### 4.1. List messages in folder

Input parameter	Constant	Value	Description
a	Yes	list	This value can not be changed for this call.
id	No		ID of the folder. Use one of the following IDs:  1 – inbox 2 – outbox 3 – sent 4 – trash
l	No		Start offset when listing messages. Setting this to “0” will start listing from the latest messages. Small “L” letter.
l2	No		End offset when listing messages. This is the actual number of messages that will be returned.

API call structure example:

```
data.php?api_key=[API_KEY]&a=list&id=[FOLDER_ID]&l=[START_NUM]&l2=[END_NUM]
```

API examples:

- List last 100 messages from inbox

```
data.php?api_key=myapikey&a=list&id=1&l=0&l2=100
```

- List last 10 messages from outbox.

```
data.php?api_key=myapikey&a=list&id=2&l=0&l2=10
```

- List 10 messages from sent folder, starting from 10<sup>th</sup> last message.

```
data.php?api_key=myapikey&a=list&id=3&l=10&l2=10
```

### Output format of the API call:

```
[MESSAGES_NUM]\n
[FULL_MESSAGES_LISTING]
```

MESSAGES\_NUM – number of returned messages.

FULL\_MESSAGES\_LISTING – listing of the messages, one per line, in following format:

```
[MSG_ID][MSG_READ][MSG_DATE][MSG_DATE_GSM][FAILED][RECEIPENTS_NUM][PHONE_NUMBER][MSG_TEXT]\n
```

Output parameter	Type	Description
MSG_ID	integer	ID of the message in the database. This number must be used to execute any API calls for the message.
MSG_READ	Boolean (1 or 0)	Read or unread message flag. This value equals to 1 if message was marked as read. Otherwise it's equals to zero.
MSG_DATE	integer	Date of the message in UNIX timestamp format.
MSG_DATE_GSM	string	Date of the message reported by GSM provider. Timezone can be different from one provider to another. It is recommended to use MSG_DATE value for date operations.
FAILED	Boolean (1 or 0)	Failed flag. If set to 1 message was not sent to some recipients. Otherwise it's always 0.
RECEIPENTS_NUM	integer	Number of recipients for the message.
PHONE_NUMBER	string	Main phone number of the message. In case there is more than one recipient, message info must be requested to get all other phone numbers. In case this is incoming message, sender number will be displayed.
MSG_TEXT	string	ASCII text of the message.

**4.2. Show message info**  
(list all recipients and message status)

Input parameter	Constant	Value	Description
a	Yes	msg_info	This value can not be changed for this call.
id	No		ID of the message in the database (number), retrieved from folder listing

API call structure example:

```
data.php?api_key=[API_KEY]&a=msg_info&id=[MSG_ID]
```

API examples:

- Show info about message ID 75

```
data.php?api_key=myapikey&a=msg_info&id=75
```

**Output format of the API call:**

```
[MESSAGES_STATUS_INFO]\n\n
[RECEIPENTS_LIST]
```

MESSAGES\_STATUS\_INFO – status of the message in the following format:

```
Status:[MSG_STATUS]\n
Recipients:[RECEIPENTS_NUMBER]\n
Scheduled:[SCHEDULED_DATE]\n
```

Output parameter	Type	Description
MSG_STATUS	string	<ul style="list-style-type: none"> <li>received - message received to inbox</li> <li>pending - message is pending to be sent</li> <li>sent - message sent</li> <li>sent with errors - message sent with some errors, check recipients list for details</li> </ul>
RECEIPENTS_NUMBER	integer	<ul style="list-style-type: none"> <li>number of recipients in the list below.</li> <li>for incoming message this number will be zero, and no recipients list will be displayed.</li> </ul>
SCHEDULED_DATE	integer	<ul style="list-style-type: none"> <li>Unix timestamp when scheduled message will be sent, this line only appear when submitted message was scheduled for future delivery. Otherwise this line will not appear.</li> </ul>

RECEIPENTS\_LIST – list of recipients phone numbers, one per line, and status of the delivery.

[PHONE\_NUMBER][STATUS]\n

Output parameter	Type	Description
PHONE_NUMBER	string	phone number of the recipient or sender
STATUS	string	status of the delivery (sent / not sent)

Note that phone number separated from status by | character.

### 4.3. Set message as read

Input parameter	Constant	Value	Description
a	Yes	set_read	This value can not be changed for this call.
id	No		ID of the message in the database (number), retrieved from folder listing
folder_id	No		ID of the folder. Use one of the following IDs:  1 – inbox 2 – outbox 3 – sent 4 – trash

API call structure example:

```
data.php?api_key=[API_KEY]&a=set_read&id=[MSG_ID]&folder_id=[FOLDER_ID]
```

API examples:

- Set message ID 75 as read in the Inbox folder.

```
data.php?api_key=myapikey&a=set_read&id=75&folder_id=1
```

#### Output format of the API call:

Output from this call is Boolean 1 or 0.

1 – message marked as read

0 – message not marked as read (in case it's already read output will be 0)

1

**Note:** FOLDER\_ID is required for proper folder counts, if it's not supplied API call will work but correct read/unread folder count will be lost. Always supply FOLDER\_ID for proper operation.

#### 4.4. Move message to another folder or delete message

Input parameter	Constant	Value	Description
a	Yes	move_msg	This value can not be changed for this call.
id	No		ID of the message in the database (number), retrieved from folder listing or array of numbers (string) in the following format: ID1_ID2_ID3_IDn
from_id	No		ID of the folder. Use one of the following IDs:  1 – inbox 2 – outbox 3 – sent 4 – trash
to_id	No		ID of the folder. Use one of the following IDs:  1 – inbox 2 – outbox 3 – sent 4 – trash

API call structure example:

```
data.php?api_key=[API_KEY] &a=move_msg&id=[MSG_ID]&from_id=[FROM_FOLDER_ID] &to_id=[TO_FOLDER_ID]
```

API examples:

- Move message ID 15 from Inbox to Trash:

```
data.php?api_key=myapikey&a=move_msg&id=15&from_id=1&to_id=4
```

- Delete message ID 15 permanently from Trash:

```
data.php?api_key=myapikey&a=move_msg&id=15&from_id=4&to_id=4
```

- Move two messages ID 5 and ID 6 from inbox to Trash:

```
data.php?api_key=myapikey&a=move_msg&id=5_6&from_id=1&to_id=4
```

- Delete two messages ID 5 and ID 6 permanently from Trash:

```
data.php?api_key=myapikey&a=move_msg&id=5_6&from_id=4&to_id=4
```

#### Output format of the API call:

1 – message moved successfully

Any other output should be considered as failed result.

```
1
```

**Note:** It is possible to execute this call twice or more times for the same message with successful result output. This should be avoided as folder message count will be broken. Only move message once from one folder to another.

## 4.5. Send message

Input parameter	Constant	Value	Description
a	Yes	send_msg	This value can not be changed for this call.
delay	No		Unix timestamp when this message must be sent. Use empty string to send without delay. In case you supply timestamp in the past, message will be sent without delay.
request_report	No		1 or 0. Set to 1 to request delivery report from GSM provider. Set to 0 not to request delivery report.
to	No		List of phone numbers separated with semi-column ";". Message will be sent to these phone numbers. This field must be URL encoded.
txt	No		Message text. In case of message text if more than maximum allowed message size, message will be send as multipart message. This field must be URL encoded.

API call structure example:

```
api_key=[API_KEY] &a=send_msg&delay=[DELAY_TIMESTAMP]&request_report=[DELIVERY_REPORT]&to=[PHONE_NUMBERS]&txt=[MSG_TEXT]
```

API examples:

- Send message without delay and without delivery report to the numbers +4369911070350 and +4369911070351 with the text "test message":

```
data.php?api_key=myapikey&a=send_msg&delay=&request_report=0&to=4369911070350;4369911070351&txt=test+message
```

## Output format of the API call:

Successfully sent message:

```
RECEIPIENTS_NUMBER\n\nPHONE_NUMBERS
```

RECEIPIENTS\_NUMBER (number) – number of recipients for inserted message.

PHONE\_NUMBERS – list of successfully inserted phone numbers for this message in the following format:

```
PHONE_NUMBER_1\nPHONE_NUMBER_2\nPHONE_NUMBER_n
```

Unsuccessfully sent message:

Error message will be displayed. In case first line of the output value is not numeric and not greater than zero, message sending must be considered as failed.

## 5. Example code

### 5.1. PHP code samples

Following examples demonstrate how to interact with Messagenie API via PHP scripts

- Get messages from inbox and move them to Trash.  
This code can be run constantly to receive new messages from Inbox into your application or database.  
Note: this code does not handle setting messages read flag, in case you need correct counting displayed at standard Messagenie interface, you need to set message read flag for each message, before moving messages to Trash.

```
<?
$host = "192.168.1.150";
$api_key = "myapikey";
$request = "/data.php?api_key=$api_key&a=list&id=1&l=0&l2=100";
$response = "";

//connect to Messagenie host
$fp = fsockopen($host, 80, $errno, $errstr, 30);
if (!$fp) {
    echo "Error: $errstr ($errno)<br />\n";
} else {
    $out = "GET $request HTTP/1.0\r\n";
    $out .= "Host: $host\r\n";
    $out .= "Connection: Close\r\n\r\n";

    //send API request
    fwrite($fp, $out);

    while (!feof($fp))
    {
        $response .= fgets($fp, 128);
    }

    //close connection
    fclose($fp);
}

//remove server headers and parse response
$arr = explode("\r\n\r\n",$response);
```

```

if(count($arr) > 1)
{
$response = trim($arr[1]);

if($response != "")
{
//parsing response
if($response == 0)
{
//no messages in the list
echo "No messages\n";
}
else
{
//some messages present
$trash_ids = "";
$arr = explode("\n",$response);
for($i=0;$i<count($arr);$i++)
{
//parsing every line of response
$line = $arr[$i];

if($i==0)
{
//first line is the number of messages
echo "Received $line message(s)\n";
}
else
{
//array containing message data according to API docs
$arr_msg = explode('|',$line);

$msg_id = $arr_msg[0];
$msg_read = $arr_msg[1];
$msg_date = $arr_msg[2];
$msg_date_gsm = $arr_msg[3];
$msg_failed = $arr_msg[4];
$msg_receipients_num = $arr_msg[5];
$msg_phone_number = $arr_msg[6];

//the rest is msg text, even is txt contains |
if(count($arr_msg) > 7)
{
$msg_txt = "";
for($k=7;$k<count($arr_msg);$k++)
{
if($msg_txt != "") { $msg_txt .= '|'; }
$msg_txt .= $arr_msg[$k];
}
}
}
}
}
}

```

```

else
{
$msgs_txt = $arr_msg[7];
}

if(($msg_id > 0) && ($msg_date > 0))
{
//building IDs for further API call to move to trash
if($trash_ids != "") { $trash_ids .= '_'; }
$trash_ids .= $msg_id;

//display message phone and text
echo "Received message from $msg_phone_number. Text: [$msgs_txt]\n";

//process messages with your application here

}

}

} //end of for loop

//moving displayed messages to Trash folder so they will not be displayed on next API call
echo "Moving following message IDs to Trash folder: $trash_ids\n";

//generating API call again
$request = "/data.php?api_key=$api_key&a=move_msg&id=$trash_ids&from_id=1&to_id=4";
$response = "";

//connect to Messagenie host
$fp = fsockopen($host, 80, $errno, $errstr, 30);
if (!$fp) {
echo "Error: $errstr ($errno)<br />\n";
} else {
$out = "GET $request HTTP/1.0\r\n";
$out .= "Host: $host\r\n";
$out .= "Connection: Close\r\n\r\n";

//send API request
fwrite($fp, $out);

while (!feof($fp))
{
$response .= fgets($fp, 128);
}

//close connection
fclose($fp);

```

```
    }

    $arr = explode("\r\n\r\n",$response);
    if(count($arr) > 1)
    {
        $response = trim($arr[1]);
        if($response != "")
        {
            if($response == 1)
            {
                echo "Messages moved to Trash\n";
            }
            else
            {
                echo "Some errors when moving messages:\n$response\n";
            }
        }
    }

}

}
else
{
    echo "Error while receiving messages.";
}
}

?>
```

- Send new message

```
<?
$host = "192.168.1.150";
$sapi_key = "myapikey";

//list of phone numbers
$phone_numbers = '4369911070350;4369911070351';

//message text
$txt = 'This is test message sent through API call';

//url encode txt and phones
$txt = urlencode($txt);
$phone_numbers = urlencode($phone_numbers);
```

```

//actual GET request to be sent
$request =
"/data.php?api_key=$api_key&a=send_msg&delay=&request_report=1&to=$phone_numbers&txt=$txt";

$response = "";

//connect to Messagenie host
$fp = fsockopen($host, 80, $errno, $errstr, 30);
if (!$fp) {
    echo "Error: $errstr ($errno)<br />\n";
} else {
    $out = "GET $request HTTP/1.0\r\n";
    $out .= "Host: $host\r\n";
    $out .= "Connection: Close\r\n\r\n";

    //send API request
    fwrite($fp, $out);

    while (!feof($fp))
    {
        $response .= fgets($fp, 128);
    }

    //close connection
    fclose($fp);
}

//remove server headers and parse response
$arr = explode("\r\n\r\n",$response);
if(count($arr) > 1)
{
    $response = trim($arr[1]);
    if($response != "")
    {
        //parsing response
        $arr = explode("\n\n",$response);
        if(count($arr) > 1)
        {
            $count = $arr[0];
            $phones = $arr[1];

            if($count > 0)
            {
                echo "Message sent successfully\n";
                echo "Number of recipients: $count\n";
                echo "Phone numbers:\n$phones";
            }
            else
            {
                echo "Error while sending message.\n";
            }
        }
    }
}

```

```
    echo "$response";
  }
}
else
{
  echo "Error while sending message.\n";
  echo "$response";
}
}
else
{
  echo "Error while sending messages.";
}
}
?>
```

## 5. Revision history

Date	Version	Notes
03.12.2010	1.1	Updated PHP samples code and note about read messages count.
01.12.2010	1.0	Initial release.