Integración HTTP REST

Versión 2.0

Índice

Introducción

La plataforma REST Gateway permite al usuario enviar mensajes a través de HTTP o HTTPS de una manera sencilla y rápida, pudiendo enviar mas de 500 mensajes en una sola petición. Para poder acceder a sus estadísticas y datos de facturación puede acceder a la web **WauSMS** con sus datos de usuario.

Esta documentación describe los parámetros necesarios y opcionales para utilizar todas las posibilidades para el envío de mensajes sms siguiendo las especificaciones REST. Tanto las peticiones como las respuestas de la API REST están en formato JSON, haciendo muy sencillo la utilización del la API con cualquier lenguaje de programación .

PLATAFORMA TÉCNICA

Petición de envío de SMS

Cada petición que se realice tendrá que incluir en la cabecera de la petición http la autenticación del cliente. Para ello se utiliza la autenticación de acceso básica de HTTP.

La cabecera de autorización se construye combinando la cadena “usuario: contraseña” y codificándola en base64. A esta cadena se antepone la cadena “Authorization: Basic”

Por ejemplo, para el usuario “miuser” y el password “mipass” la cabecera resultante sería: **Authorization: Basic bWl1c2VyOm1pcGFzcw==**

A continuación se detallará las opciones de envío disponibles, la URL a la que se debe llamar, y los parámetros que admite.

Para generar la URL el cliente deberá hacer una llamada POST a la siguiente dirección: **https://dashboard.wausms.com/Api/rest/message**

Petición JSON:

Ejemplo de petición básica: {"to":["34666555444"],"text":"mensaje de texto","from":"msg"}

Parámetros posibles:

- **text:** Texto del mensaje. Como máximo puede tener 160 caracteres si no se especifica que el mensaje sea multiparte (ver parámetro "parts"). El texto tiene que estar codificado en UTF-8
- **to:** número de teléfono móvil destinatario del mensaje. Debe incluir el prefijo (Ej: En España 34666666666). Este campo permite indicar multiples destinatarios, para ello hay que incluir todos los números en un array.
- **from:** Texto del Remitente, esta etiqueta se compondrá de 15 números o 11 caracteres alfanuméricos.
- **coding (opcional):** Los posibles valores son "gsm", "gsm-pt" y "utf-16". El valor por defecto es "gsm". El número máximo de caracteres para mensajes normales es de 160 para la codificación GSM7, 155 para la codificación GSM-PT y 70 para la codifcación UCS2 (UTF16). El número máximo de caracteres para mensajes concatenados es de 155 para la codificación GSM7, 149 para la codificación GSM-PT y 67 para la codifcación UCS2 (UTF16).
- **fSend (opcional):** Fecha de envío del mensaje. Si se necesita enviar mensajes programados se puede especificar la fecha de envío indicando la fecha en formato YYYYmmddHHiiss (Ej: 20130215142000 sería el 15 de febrero de 2013 a las 14:20 UTC). La fecha se tiene que especificar en tiempo UTC (GMT+0). No se podrán programar envíos mas tarde de los 30 días siguientes. En caso de envío inmediato no se tiene que especificar este parámetro.
- **parts (opcional):** Indica el número máximo de partes en que se dividirá el mensaje para su envío. Esta variable tiene valor 1 por defecto, por lo que si no se especifica y se envía un mensaje de mas de 160 caracteres para codificación 0, el mensaje fallará. Hay que tener en cuenta que los mensajes concatenados solo pueden tener 153 caracteres por parte y que cada parte se tarifica como un envío. El servidor sólo utilizará el mínimo de partes necesarias para realizar el envío del mensaje, aunque el número de partes especificado sea superior al necesario. Si el número de partes especificado es inferior al necesario para el envío, éste fallará con el error 105.
- **trsec (opcional):** Los valores posibles son 1 y 0. Con el valor 0 el servidor no modifica ningún carácter del mensaje, este es el valor por defecto. Con el valor 1 el servidor se encarga de modificar los caracteres comunes no validos en GSM7 a caracteres validos con la siguiente tabla de traducción: "á"=>"a", "í"=>"i", "ó"=>"o", "ú"=>"u", "ç"=>"Ç", "Á"=>"A", "Í"=>"I", "Ó"=>"O", "Ú"=>"U", "À"=>"A", "È"=>"E", "Ì"=>"I", "Ò"=>"O", "Ù"=>"U", "o"=>"", "a"=>"", "Õ"=>"O", "õ"=>"o", "â"=>"a", "ê"=>"e", "î"=>"i", "ô"=>"o", "û"=>"u", "Â"=>"A", "Ê"=>"E", "Î"=>"I", "Ô"=>"O", "Û"=>"U", "ã"=>"a", "Ã"=>"A".
- **reference (opcional):** Referencia del envío. Si no se especifica, se generará una referencia automáticamente cada mes con la siguiente nomenclatura: API_SMS_yyyy_mm siendo yyyy el año actual y mm el mes actual.
- **tags (opcional):** Array de tags. Ej: ["tag1","tag2"]

Ejemplo de petición CURL:

```
curl -X POST \
-H "Content-Type: application/json" \
-H "Accept: application/json" \
-H "Authorization: Basic bWl1c2VyOm1pcGFzcw==" \
-d "{\"to\":[\"34666555444\"],\"text\":\"mensaje de texto\",\"from\":\"msg\"}" \
https://dashboard.wausms.com/Api/rest/message
```

Ejemplo de petición PHP:

```php
<?php
	$post['to'] = array('34666555444');
	$post['text'] = "mensaje de texto";
	$post['from'] = "msg";
	$user ="miuser";
	$password = 'mipass';
	$ch = curl_init();
	curl_setopt($ch, CURLOPT_URL, "https://dashboard.wausms.com/Api/rest/message");
	curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
	curl_setopt($ch, CURLOPT_POST, 1);
	curl_setopt($ch, CURLOPT_POSTFIELDS,    json_encode($post));
	curl_setopt($ch, CURLOPT_HTTPHEADER,
	array(
		"Accept: application/json",
		"Authorization: Basic ".base64_encode($user.":".$password)));
	$result = curl_exec ($ch);
?>
```

La clave de acceso (password) y el código del cliente (username) serán proporcionados por la empresa. Hay que comentar que con objeto de aumentar la seguridad del sistema, el cliente deberá indicar la IP desde donde se va a conectar, solo se permitirán envíos de la IP indicada por el cliente.

Códigos de estado de respuestas

La API REST puede responder con los estados HTTP siguientes:

Código estado	Descripción	Detalles
202	Accepted	El mensaje se ha aceptado para su posterior proceso
207	Multi-status	El mensaje se ha aceptado para su posterior proceso, pero algunos de los destinatarios son incorrectos.
400	Bad request	La petición contiene errores, el mensaje no ha sido aceptado
401	Unauthorized	Fallo en la autenticación del cliente
402	Payment required	El cliente no dispone de saldo suficiente
500	Internal server error	El servidor ha tenido un error interno

En el cuerpo de la respuesta HTTP se entrega un JSON con los detalles del resultado, estas son las respuestas posibles:

Código estado 202:

[{"accepted":true,"to":"34666555444","id":"102648819"}]

Código estado 207:

[{"accepted":true,"to":"34666555444","id":"102648819"}]

Código estado 202:

[{"accepted":true,"to":"34626690739","id":"102648820"},{"accepted":false,
"to":"34","error":{"code":102,"description":"No valid recipients"}}]

Código estado 400:

{"error":{"code":102,"description":"No valid recipients"}}
{"error":{"code":104,"description":"Text message missing"}}
{"error":{"code":105,"description":"Text message too long"}}
{"error":{"code":106,"description":"Sender missing"}}
{"error":{"code":107,"description":"Sender too long"}}
{"error":{"code":108,"description":"No valid Datetime for send"}}
{"error":{"code":109,"description":"Notification URL incorrect"}}
{"error":{"code":110,"description":"Exceeded maximum parts allowed or incorrect number of parts"}}
{"error":{"code":113,"description":"Invalid coding"}}

Código estado 401:

{"error":{"code":103,"description":"Username or password unknown"}}
{"error":{"code":111,"description":"Not enough credits"}}

Código estado 402:

{"error":{"code":111,"description":"Not enough credits"}}

Anexo A: Acuses de recibo

Si se desean recibir los acuses de recibo en tiempo real se deberá especificar la variable “dlr-url” con la URL del cliente donde quiere que se notifique el estado del envío.

El funcionamiento consiste en especificar en cada petición http la URL donde se desea que realice una petición de nuestro servidor cuando se reciba una notificación por parte de la operadora. Para ello el cliente debe disponer de un servidor http capaz de recibir esas notificaciones.

Nuestro servidor enviara las variables por el método GET tal como el cliente quiera, para ello en la URL que nos envía tiene que poner el nombre de la variable seguido de un carácter de escape que contendrá el valor, los caracteres de escape tienen la forma del carácter “%” seguido de una letra. Este seria un ejemplo de URL: **http://mi.server.com/notifica.php?remitente=%p&tel=%P&estado=%d**

Estos son los caracteres de escape definidos:

- **%i** Identificador de **WauSMS** que se entregó cuando se hizo el envío.
- **%d** Valor del acuse de recibo.
- **%p** El remitente del SMS.
- **%P** El número de teléfono del receptor del mensaje SMS.
- **%t** Fecha del envío del mensaje con formato "YYYY-MM-DD HH:MM", e.j., "2015-09-21 14:18".
- **%c** coste del mensaje.
- **%s** estado (REJECTD, DELIVRD, EXPIRED, DELETED, UNDELIV, ACCEPTD, UNKNOWN, RECEIVED).
- **%y** fecha del DLR del mensaje con formato "YYYY-MM-DD HH:MM", e.j., "2020-09-21 14:19".
- **%n** número de parte (mensajes concatenados).

El valor %d es el que nos devolverá el estado final del envío, los valores posibles son:

- **1:** El mensaje ha sido entregado al destinatario.
- **2:** El mensaje no se ha podido entregar al destinatario.
- **4:** El mensaje ha sido entregado al SMSC, es una notificación intermedia, no un resultado final
- **16:** No se ha podido entregar a la operadora final

Para explicar mejor el proceso, a continuación se da un ejemplo de cómo seria el envío de un sms y la recepción de su acuse de recibo.

En primer lugar enviamos el sms con la variable dlr-url donde indicaremos la URL donde queremos recibir la notificación de entrega, añadiremos a esta URL nuestro identificador de envío para poder identificar inequívocamente cuando lo recibamos. La url final para la notificación seria: **http://mi.server.com/notifica.php?idenvio=123&remitente=%p&tel=%P&estado=%d**

Ejemplo de petición CURL:

```
curl -X POST \
-H "Content-Type: application/json" \
-H "Accept: application/json" \
-H "Authorization: Basic bWl1c2VyOm1pcGFzcw==" \
-d "{\"to\":[\"34666555444\"],\"text\":\"mensaje \",\"from\":\"msg\",\"dlr-
url\":\"http://mi.server.com/notifica.php?remitente=%p&tel=%P&estado=%d\"}" \
https://dashboard.wausms.com/Api/rest/message
```

Ejemplo de petición PHP:

```
<?php
    $post['to'] = array('34666555444');
    $post['text'] = "mensaje de texto";
    $post['from'] = "msg";
    $post
```

```
['dlr-url'] = "http://mi.server.com/notifica.php?idenvio=7584&remitente=%p&tel=%P&estado=%d";
	$user ="miuser";
	$password = 'mipass';
	$ch = curl_init();
	curl_setopt($ch, CURLOPT_URL, "https://dashboard.wausms.com/Api/rest/message");
	curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
	curl_setopt($ch, CURLOPT_POST, 1);
	curl_setopt($ch, CURLOPT_POSTFIELDS,     json_encode($post));
	curl_setopt($ch, CURLOPT_HTTPHEADER,
	array(
		"Accept: application/json",
		"Authorization: Basic ".base64_encode($user.":".$password)));
	$result = curl_exec ($ch);
?>
```

Suponiendo que todos los mensajes puedan ser entregados, recibiremos al script notifica.php tres peticiones con el estado=1, remitente=TEST, idenvio=7584 y el número de teléfono correspondiente.

Anexo B: Conjunto de caracteres GSM7

Conjunto de caracteres básico

	0x00	0x10	0x20	0x30	0x40	0x50	0x60	0x70
0x00	@	?	SP	0	¡	P	¿	p
0x01	£	_	!	1	A	Q	a	q
0x02	$	?	"	2	B	B	b	r
0x03	¥	?	#	3	C	S	c	s
0x04	è	?	¤	4	D	T	d	t
0x05	é	?	%	5	E	U	e	u
0x06	ù	?	?	6	F	V	f	v
0x07	ì	?	'	7	G	W	g	w
0x08	ò	?	(	8	H	X	h	x
0x09	Ç	?	)	9	I	Y	i	y
0x0A	LF	?	*	:	J	Z	j	z
0x0B	Ø	ESC	+	;	K	Ä	k	ä
0x0C	ø	Æ	,	<	L	Ö	l	ö
0x0D	CR	æ	-	=	M	Ñ	m	ñ
0x0E	Å	ß	.	>	N	Ü	n	ü
0x0F	å	É	/	?	O	§	o	à

Extensión del conjunto de caracteres básico, estos caracteres ocupan dos posiciones

	0x00	0x10	0x20	0x30	0x40	0x50	0x60	0x70
0x00					|			
0x01								
0x02								
0x03								
0x04		^						
0x05							€	

	0x00	0x10	0x20	0x30	0x40	0x50	0x60	0x70
0x06								
0x07								
0x08			}					
0x09			{					
0x0A	FF							
0x0B		SS2						
0x0C				[				
0x0D	CR2			~				
0x0E				]				
0x0F			\					