Integração HTTP REST

Versão 2.0

Índice

Introdução

A Plataforma REST Gateway permite ao usuário enviar mensagens através de HTTP ou HTTPS de forma simples e rápida, podendo enviar mais de 500 mensagens em uma única solicitação. Para poder ter acesso as suas estatísticas e dados de faturamento, acesse a página web **WauSMS** com os seus dados de usuario.

Esta documentação descreve os parâmetros necessários e facultativos para a utilização de todas as possibilidades no envio de mensagens SMS, seguindo as especificações REST. Tanto as solicitações como a as respostas da API REST estão em formato JSON, tornando fácil a utilização da API com qualquer linguagem de programação .

PLATAFORMA TÉCNICA

Solicitação de envio de SMS

Cada solicitação que se realize, deverá incluir no cabeçalho da solicitação HTTP a autenticação do cliente. Para tal utiliza-se a autenticação de acesso básico do HTTP.

O cabeçalho de autorização é formado pela combinação "usuário:password" (sem aspas) devendo ser codificados em base64. O hash gerado, deverá ser utilizado em "Authorization: Basic"

Por exemplo, para o usuário "miuser" e a password "mipass" o cabeçalho resultante seria: **Authorization: Basic bWl1c2VyOm1pcGFzcw==**

Em seguida, detalhamos as opções de envio disponíveis a URL ao qual se deverá chamar, assim como, os parâmetros suportados.

Para gerar a URL o cliente deverá fazer uma chamada POST ao seguinte endereço: **https://dashboard.wausms.com/Api/rest/message**

Solicitação JSON:

Exemplo de solicitação básica: {"to":["5511966554433"],"text":"teste de envio api rest","from":"teste"}

Possíveis parâmetros:

- **text:** Texto da mensagem. Poderá conter no máximo 160 caracteres caso não se especifique que a mensagem é multiparte (veja parâmetro "parts"). O texto deve ser codificado em UTF-8
- **to:** número de telefone do destinatário da mensagem. Deve-se incluir o prefixo (Exemplo: No Brasil 5511966554433). Este parâmetro, permite a indicação de múltiplos destinatários. Para isso, inclua todos os números numa matriz/array.
- **from:** Remetente da mensagem. Nesse parâmetro pode ser utilizado 15 caracteres numéricos ou 11 alfanuméricos.
- **coding (opcional):** Los posibles valores son "gsm", "gsm-pt" y "utf-16". El valor por defecto es "gsm". O número máximo de caracteres para mensagens normais é 160 para a codificação GSM7, 155 para a codificação GSM-PT e 70 para a codificação UCS2 (UTF16). O número máximo de caracteres para mensagens concatenadas é 155 para a codificação GSM7, 149 para a codificação GSM-PT e 67 para a codificação UCS2 (UTF16).
- **fSend (opcional):** Data de agendamento da mensagem. Para agendar o envio da mensagem pode-se especificar a data desejada no formato YYYYmmddHHiiss (Exemplo: 20130215142000 - sendo 15 de fevereiro de 2013 às 14:20 UTC). A data deve ser especificada na hora UTC (GMT + 0). Não são permitidos agendamentos com mais de 30 dias a partir da data atual. No caso de envio imediato não é necessário especificar este parâmetro.
- **parts (opcional):** Indica o número máximo de partes em que se dividirá a mensagem ao ser enviada. Esta variável é definida por padrão com valor 1, portanto se não for especificado, ao enviar uma mensagem com mais de 160 caracteres para codificação "GSM", a mensagem irá falhar. Tenha em mente que as mensagens concatenadas podem ter apenas 153 caracteres cada uma. Cada uma das partes é tarifada como um envio. O servidor só utilizará o mínimo de partes necessárias para enviar as mensagens de texto, mesmo que o número especificado de partes seja maior do que o necessário. No caso em que o número de partes seja menor do que o necessário para enviar mensagens de texto, o envio falhará com o erro 105.
- **trsec (opcional):** Os valores permitidos são 1 e 0. Com o valor "0" o servidor não modifica quaisquer caracteres da mensagem, este é o valor padrão. Com o valor "1" o servidor se encarrega de transformar os caracteres comuns inválidos no GSM7 em caracteres válidos com a seguinte tabela de tradução: "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):** Referência de envio. Se não especificado, uma referência será gerada automaticamente a cada mês com a seguinte nomenclatura: API_SMS_yyyy_mm em que yyyy é o ano atual e mm o mês atual.
- **tags (opcional):** Array de tags. Exemplo: ["tag1","tag2"]

Exemplo de requisição CURL:

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

Exemplo de requisição PHP:

```php
<?php
	$post['to'] = array('34666555444');
	$post['text'] = "mensagem 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);
?>
```

A senha (password) e o código do cliente (username) serão fornecidos pela empresa. Devemos mencionar que, a fim de melhorar a segurança do sistema, o cliente deverá indicar o IP de onde se conectará, somente serão permitidos envios provenientes do IP especificado pelo cliente.

Códigos do status de resposta

A API REST pode responder com os seguintes estados HTTP:

Código do estado	Descrição	Detalhes
202	Accepted	A mensagem foi aceita para seu processo posterior
207	Multi-status	A mensagem foi aceita para seu processo posterior, mas alguns destinatários estão incorretos.
400	Bad request	A solicitação contém erros, a mensagem não foi aceita
401	Unauthorized	Falha na autenticação do cliente
402	Payment required	O cliente não tem saldo suficiente
500	Internal server error	O servidor teve um erro interno

No corpo da resposta HTTP se entrega um JSON com os detalhes do resultado, estas são as possíveis respostas:

Código do estado 202:

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

Código do estado 207:

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

Código do estado 202:

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

Código do 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 do estado 401:

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

Código do estado 402:

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

Anexo A: Avisos de recebimento

Se você desejar receber confirmações em tempo real, deve-se especificar a variável "dlr-url" com a URL do cliente onde quer que a notificação de status seja entregue.

O funcionamento consiste em especificar em cada solicitação HTTP o URL onde deseja-se que seja realizada uma solicitação do nosso servidor ao receber uma notificação por parte da operadora. Para tal o cliente deve ter um servidor http capaz de receber tais notificações.

O nosso servidor enviará as variáveis pelo método GET da maneira que queira o cliente, por isso no URL que nos envie, é importante indicar o nome da variável seguido por um caractere de escape que conterá o valor. Os caracteres de escape têm a forma do caractere "%", seguido de uma letra. Este seria um exemplo de URL: **http://mi.server.com/notifica.php?remitente=%p&tel=%P&estado=%d**

Estes são os caracteres de escape definidos:

- **%i** Identificador de **WauSMS** que foi entregue ao realizar o envio.
- **%d** Valor do aviso de recebimento.
- **%p** O remetente do SMS.
- **%P** O número do telefone do destinatário do SMS.
- **%t** Data do envio de mensagem com formato "YYYY-MM-DD HH:MM", por exemplo, "2015-09-21 14:18".
- **%c** custo da mensagem.
- **%s** estado (REJECTD, DELIVRD, EXPIRED, DELETED, UNDELIV, ACCEPTD, UNKNOWN, RECEIVED).
- **%y** data DLR da mensagem formatada "YYYY-MM-DD HH:MM", ex., "2020-09-21 14:19".
- **%n** número da peça (mensagens concatenadas).

O valor %d é o que nos devolverá o estado final do envio, os valores possíveis são:

- **1:** A mensagem foi entregue ao destinatário.
- **2:** A mensagem não pôde ser entregue ao destinatário.
- **4:** A mensagem foi entregue ao SMSC, é uma notificação intermediária, e não um resultado final
- **16:** Não foi possível entregar a operadora final

Para explicar melhor o processo, a seguir, apresentamos um exemplo de como seria o envio de um sms e a recepção de seu aviso de recebimento.

Primeiro vamos enviar o sms com a variável dlr-url onde indicaremos o URL onde queremos receber o aviso de entrega, adicionaremos a este URL nosso identificador de envio para poder identificá-lo inequivocamente ao recebê-lo. O URL final para a notificação seria: **http://mi.server.com/notifica.php?idenvio=123&remitente=%p&tel=%P&estado=%d**

Exemplo de requisição CURL:

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

Exemplo de requisição PHP:

```
<?php
    $post['to'] = array('34666555444');
    $post['text'] = "mensagem 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);
?>
```

Assumindo que todas as mensagens possam ser entregues, receberemos do script notifica.php três requisições com o estado= 1, remetente=TEST, idenvio = 7584 e o número do telefone correspondente.

Anexo B: Conjunto de caracteres GSM7

Conjunto de caracteres básicos

	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

Extensão do conjunto de caracteres básicos, estes caracteres ocupam duas posições

	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			\					