

MANUAL - API

Overview

Esta é a documentação de referência para a API do tipo REST da Trinity. A própria API é baseada em recursos que são representados pelo formato JSON e são manipulados usando o protocolo HTTP.

Recursos

Você pode enviar mensagens através dos canais com suporte da Trinity.

Você também pode se inscrever em eventos e recebê-los em um webhook de sua escolha. Os eventos disponíveis para cada canal são:

Mensagens: receba eventos de mensagens para mensagens enviadas e / ou recebidas.

Status de mensagens: receba atualizações de status para mensagens enviadas.

Pré-requisitos

Antes de usar esta API, você precisa do seguinte:

- Conta Trinity: crie uma conta no site da plataforma Trinity
- Integrações: configure os canais desejados para enviar e / ou receber mensagens na página de integrações
- Token de API: crie um token de API no console de API

Visualize seu relatório de uso

Você também pode acessar a plataforma Trinity para visualizar suas **Estatísticas**

Métodos HTTP

Métodos HTTP são usados para manipular recursos. Porém, como nem todos os recursos permitem todas as operações HTTP, observe OK a referência de cada recurso abaixo.

Métodos usados com endpoints de coleta:

HTTP Method	Operation	Success HTTP status
GET	List collection items	200 - OK
POST	Create a new item	200 - OK

Métodos utilizados nos endpoints:

HTTP Method	Operation	Success HTTP status
GET	Retrieve one resource item	200 - OK
DELETE	Delete one resource item	204 - No content
PATCH	Update one resource item	200 - OK

Entendendo os erros

Quando ocorre um erro, a API retorna um código de status HTTP 4xx ou 5xx e o payload com o Error Object.

Códigos de erros detalhados abaixo:

Http Status Code	Code	Message	Retry request
400	VALIDATION_ERROR	Validation error	No
401	AUTHENTICATION_ERROR	No authorization token was found	No
404	NOT_FOUND	Not found	No
409	DUPLICATED	Entity already exists	No
500	INTERNAL_ERROR	Internal error	Yes

Autenticação

Token

Para usar esta API, você precisa enviar o token da API em cada solicitação. O token deve ser enviado no cabeçalho HTTP 'X-API-TOKEN'.

Exemplo: X-API-TOKEN: hKp94crjv9OF3UGrCpSXUJw1-UYHhRvLKNLt

Gere seu token no console API da plataforma Trinity seguindo abaixo:

- 1 – Acesse <https://api.trinitytec.com.br>
- 2 – Faça o login com as informações de sua conta



Trinity SMS REST API

acc_manager_mapping	Show/Hide List Operations Expand Operations
account	Show/Hide List Operations Expand Operations
activate_user	Show/Hide List Operations Expand Operations
activation_key	Show/Hide List Operations Expand Operations
agreement	Show/Hide List Operations Expand Operations
auth	Show/Hide List Operations Expand Operations

- 3 – Clique em "auth" e em seguida clique no símbolo 

A screenshot of the Trinity API console showing the details for the 'auth' endpoint. The endpoint is a GET request to '/auth'. The response class is 'Successful execution' with a status of 200. The response content type is 'application/json'. There are two query parameters: 'acc_id' and 'ip'. A red arrow points to a red exclamation mark icon in the top right corner of the endpoint details panel.

auth [Show/Hide](#) | [List Operations](#) | [Expand Operations](#)

GET /auth Authorization

Response Class (Status 200)
Successful execution

Model Example Value

```
auth_get_out {  
  token (string): JWT token  
}
```

Response Content Type [application/json](#)

Parameters

Parameter	Value	Description	Parameter Type	Data Type
acc_id	<input type="text"/>	Account ID that will be used in send_sms/bulk_send_sms	query	string
ip	<input type="text"/>	Allowed IP (0.0.0.0/0 - requires trust all addresses)	query	string

4 – Insira seus dados de autenticação de sua conta e clique em “Authorize”

Data Type
string
string
double
string

5 – Gere seu token considerando os campos:

- * ip – defina um IP válido para utilizar a aplicação ou insira *.*.* para qualquer endereço de IP;
- * lifetime – defina um valor de validade do seu token em dias; ou
- * non_exp – para token que não expira. Defina o valor como “true”

6 – Em seguida clique em “Try it out”:

auth Show/Hide | List Operations | Expand Operations

GET /auth Authorization

Response Class (Status 200)
Successful execution

Model Example Value

```
auth_get_out {  
  token (string): JWT token  
}
```

Response Content Type application/json

Parameters

Parameter	Value	Description	Parameter Type	Data Type
acc_id	<input type="text"/>	Account ID that will be used in send_sms/bulk_send_sms	query	string
ip	<input type="text" value="*.*.*"/>	Allowed IP (0.0.0.0/0 - requires trust all addresses)	query	string
lifetime	<input type="text"/>	Lifetime (in days). Min value = 0.000694(one minute). Max value = 356(one year). Default value 0.083(3) = 2 hours.	query	double
non_exp	<input type="text" value="true"/>	Create non expired token.(true - yes, false - no)	query	string
save	<input type="text"/>	Save token in database (1 - yes, 0 - no)	query	integer

Response Messages

HTTP Status Code	Reason	Response Model	Headers
400	Invalid input		
401	Authorization failed		

Enviando um SMS

Após gerar o seu token através da interface de API da Trinity, você pode enviar um SMS diretamente na plataforma seguindo os seguintes passos:

1 – Clique em “send_sms”

send_sms Show/Hide List Operations Expand Operations

POST /send_sms Send SMS

Response Class (Status 200)
Successful execution

Model | Example Value

```
send_sms_post_out {  
  dnis (string, optional): B-number,  
  message_id (string): Message ID,  
  segment_num (integer, optional): Segment number (for splitted SMS)  
}
```

Response Content Type: **application/json**

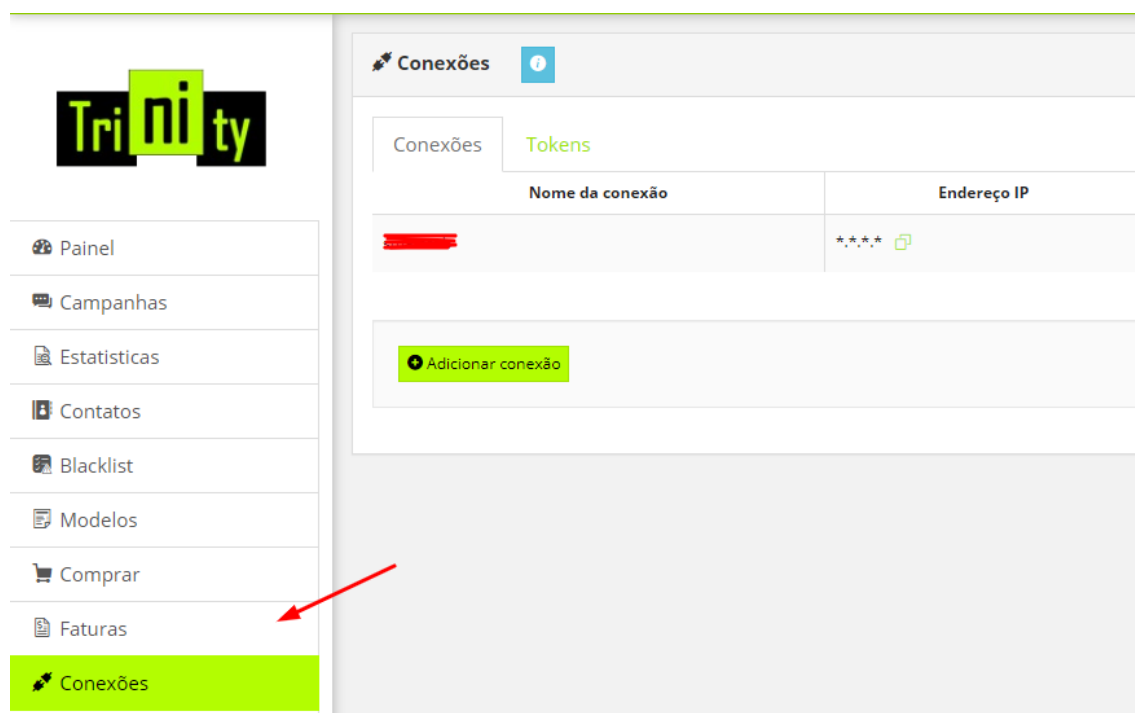
Parameters

Parameter	Value	Description	Parameter Type	Data Type
acc_id	<input type="text"/>	Account ID (for trusted connection only)	query	integer
button_action_url	<input type="text"/>	Button action url for Viber IM	query	string

2 – Insira as informações necessárias, considerando as informações destacadas

from	<input type="text" value="(required)"/>	From	query	string
im_channels	<input type="text"/>	IM channels list (comma separated, priority order)	query	string
im_message	<input type="text"/>	IM message	query	string
im_ttls	<input type="text"/>	Ttls list for IM channels (comma separated, order according to im_channels parameter)	query	string
image_url	<input type="text"/>	Button caption image url for Viber IM	query	string
message	<input type="text" value="(required)"/>	SMS message	query	string
message_label	<input type="text"/>	External label for pack of EDR	query	string
message_purpose	<input type="text"/>	Message purpose for Viber IM (transaction/promotion)	query	string
message_split_mode	<input type="text"/>	Message split mode: split, split_sar, payload, cut (default value is specified in system settings parameter "Default long message split mode")	query	string
password	<input type="text"/>	Connection authorization password	query	string
to	<input type="text" value="(required)"/>	Subscriber's phone number. Several phone numbers can be used (separated by	query	string

Através do portal Trinity, podemos gerar tokens de autenticação, autorização de IPs e criação de canais para diferentes aplicações.



Informações complementares

A partir de mensagens MT, deve-se usar nosso modelo, mas com as credenciais definidas através do portal Trinity. Abaixo um modelo de URL de callback:

```
http://1.1.1.1:8001/api?username = <username> & password = <password> & ani = <ani> & dnis = <dnis> & message = <message> & command = submit & serviceType = <serviceType> & longMessageMode = <longMessageMode>
```

As solicitações de status têm o campo especial na configuração de canais e têm seu próprio formato como abaixo:

```
http://1.1.1.1:8001/api?username = <username> & password = <password> & messageId = <messageId> & command = query
```

Os callbacks de MO devem ser configurados no Portal (se estiverem conectados via portal).



- Painel
- Campanhas
- Estatísticas
- Contatos
- Blacklist
- Modelos
- Comprar
- Faturas
- Conexões
- Configurações da conta**

MO Callback

[✎ Editar](#)

Para ativar a entrega de mensagens MO através da aplicação HTTP API você precisa configurar o link através dessa interface. Você precisa colocar a URL que contenha os seguintes marcadores:

- \$dni\$** - ID do remetente
- \$dnis\$** - Destino
- \$message\$** - mensagem de texto

Um cURL de exemplo para o envio de relatórios de entrega:

```
http://1.1.1.1:8003/api?dni=$dni$&username=SmsChannelUsername&password=SmsChannelPassword&command=entregar&dlvrM sgId=$messageID$&dlvrMsgStat=$status$
```

Um exemplo de cURL para enviar MO:

```
http://1.1.1.1:8003/api?dni=$dni$&username=SmsChannelUsername&password=SmsChannelPassword&command=mo&message=$message_text$
```