NetFortris

CommUnity

NetFortris API User Guide

CommUnity

TABLE OF CONTENTS

INTRODUCTION TO APIs

(If you are already familiar with APIs, feel free to skip the following section.)

API is an acronym for Application Programming Interface. You can think of an API as a way of structuring communication between a client (what is making a request for data) and a Server (what will respond to the request, with the requested data).

EXAMPLE

Let's say we have a Server that is in charge of providing a list of your favorite contacts. When making a request, the Server will have its own structure for how it can collect your favorite contacts list and return it to you in a response. Behind the Server, there is a database which is where the information for generating your favorite contacts is stored. The Server itself is responsible for taking a request, asking the database for the data (list of favorite contacts), then returning the requested data in a response.

In order to successfully communicate with the Server, we need to use a language structure it understands and expects.

If you were to send a request to the Server saying: "hello server, give me my favorite contacts", the Server will likely not know how to handle this request and not be able to respond. How does the Server know who is making the request, to respond with just that specific data or how to format the data in a usable format?

Now, if we know what "magic words" the server understands and can use, for handling requests, we can format our request and get a successful response: "hello server, give me the list of favorite songs for user Id 12345". The difference here is that the Server can use the unique User Id to query the database and provide that user's favorite contacts.

These definitions for how to make requests for data are what make up the various APIs. Different APIs have different request parameters that can be used, to have the Server return more specific results, or otherwise change how the Response

An example of a request for one of our APIs (/data/did/list) is:
curl -X POST "https://apiurl.com/data/did/list?customerId=1234" -H "accept: application/json" -H "api_key: 12345" -d ""

In the above example:

curl -X = a terminal command used to initiate a request (not part of the API itself)

POST = the method used to define the type of API request being made

https://apiurl.com/ = the base URL for where we are making the request to

/data/did/list = the specific API endpoint we are making the request to

?customerId=1234 = this a "query parameter" used in the body of our request. This tells our server to only give me the data back for the customer whose ID is 1234. In the URL structure, anything after a "?" is a query parameter.

-H = indicates that this is a header component parameter. The header component handles more information for structuring the API request and passing additional requirements the server has, outside the request body.

accept: application/json = indicates to the server what type of files we are working with here (JSON files).

api_key: 12345 = this is the authorization token we use to make sure that whoever is initiating the request is authorized to do so and furthermore is authorized to receive the requested data.

Data is returned.

NETFORTRIS API SITE

The following URL — https://developer.netfortris.com — will take you to our site that can be used to view and interact with some of our available APIs. You will be able to see the structure of the various APIs, their request parameters, as well as samples of Response Data structures. With the use of an API Key, you will also be able to try out the various APIs listed and gain access to actual Response Data.

API KEY

In order to generate an API Key, we will need the following information:

- Your Customer ID, or Company Name
- The email address associated with the User you want to create an API Key for.
- The IP address(es) that you will be making API requests from. This is so that we can permit traffic from the requested addresses.

Once this information has been provided, your API Key will be generated by a NetFortris representative and provided to you.

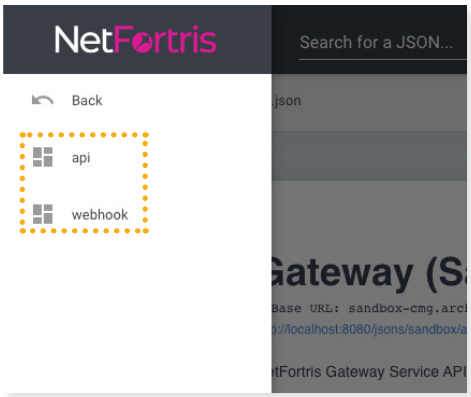
Multiple API Keys can be created for multiple users associated with your customer account if necessary.

APPLICATION OVERVIEW

NAVIGATION BAR

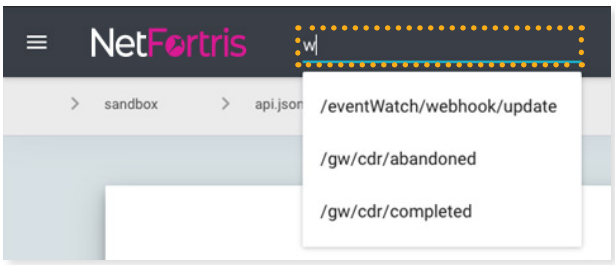
The Navigation Bar is composed of a hamburger menu which toggles a sidebar menu with available API pages, a select field for switching between environments, a search field to search for APIs listed in the available JSON files and breadcrumb links used to navigate between pages.

- Click on the hamburger menu to open the sidebar menu and select the **API/Webhook** page.



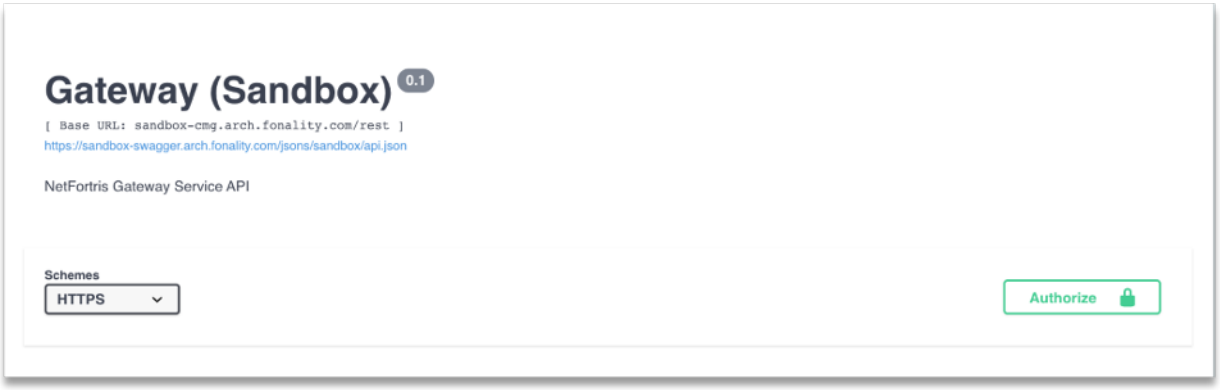
SEARCH FOR AVAILABLE APIS

Enter an item in the search field, then click on an **API route** to navigate to the page where that route is listed.



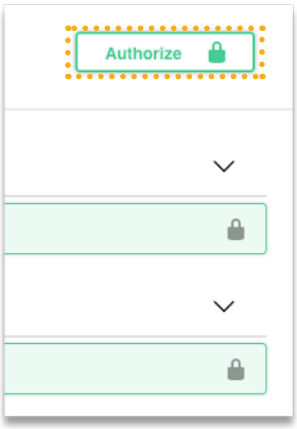
HEADER

The Header contains information about the API data endpoint for the current environment and page being displayed. This includes the base URL used to make requests, the full path to the current page API data, the request schema (http/https) and an Authorize button.



NOTE

The Schemes field defaults to HTTPS. At this time all our APIs use HTTPs so this should not be changed. Attempting to use HTTP will result in failed requests. This is only to be used in the unlikely event we need to add non-HTTPS APIs as an option in the future.



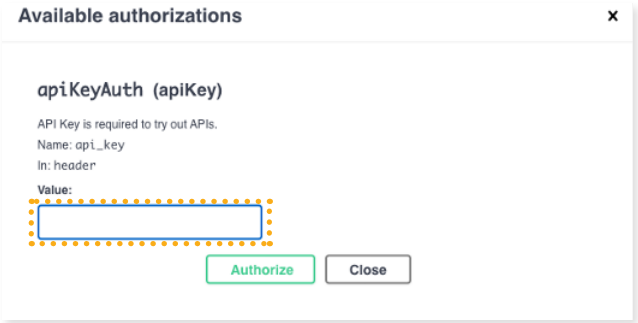
AUTHORIZATION

You will notice that the Authorize button and all API routes have a lock icon. This indicates that the current User is not authorized to "try out" any of the listed APIs. Without authorization, Users can still view the APIs, see request parameters and response schemas, however attempting to make a request will result in an unauthorized error response.

Click the **Authorize** button to open a modal .

Enter your **API Key.** (If you do not have an API Key, please see notes on obtaining your API Key previously in this document.)

Once the API Key has been added, click **Authorize**"again. Then close the modal. You will notice the lock icons will now show as unlocked.



NOTE

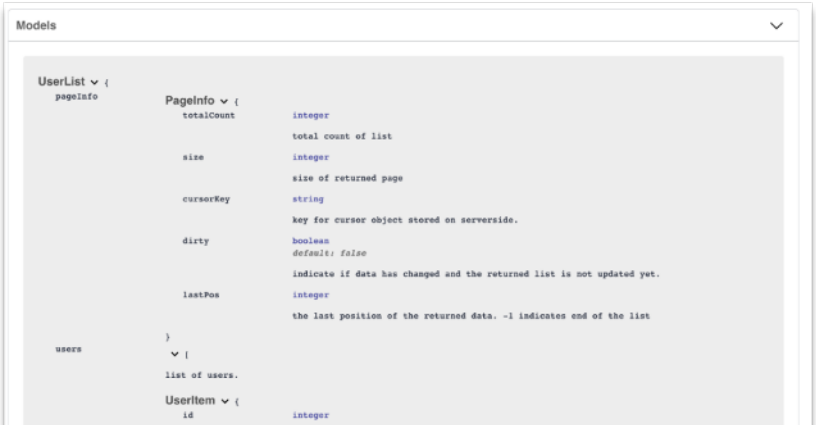
We do not store your API Key anywhere in our application. This means that anytime you navigate between API pages, (e.g., switching from Webhooks to API), you will be required to re-enter your API Key to reauthorize. Attempting to make requests with an invalid API Key will still result in unauthorized failure responses.

CommUnity

FOOTER

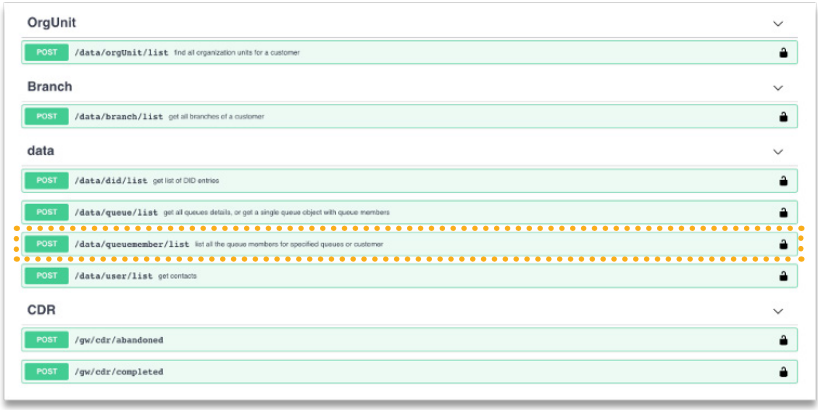
The Footer contains a list of Response object schemas that can be used to reference the structure of Response Data per API. There is also a floating menu button which will open the sidebar (just like the hamburger menu icon in the header).

Expanding the API will reveal the structure of the **Response** and **Response Data** objects.



API

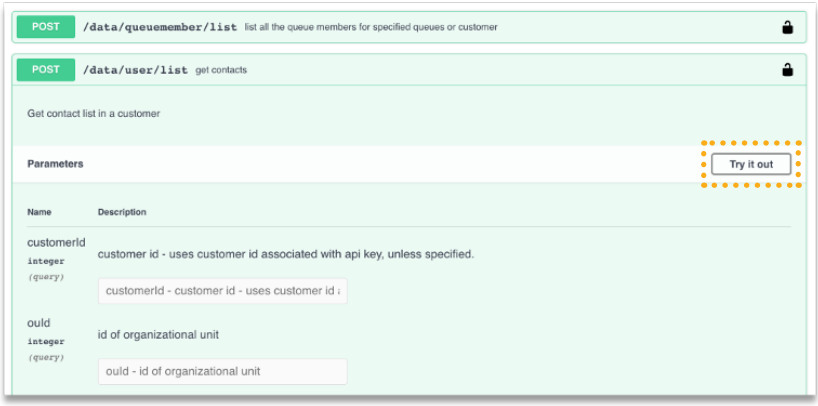
The available APIs for the currently selected page are listed in the page body, as shown on the right.



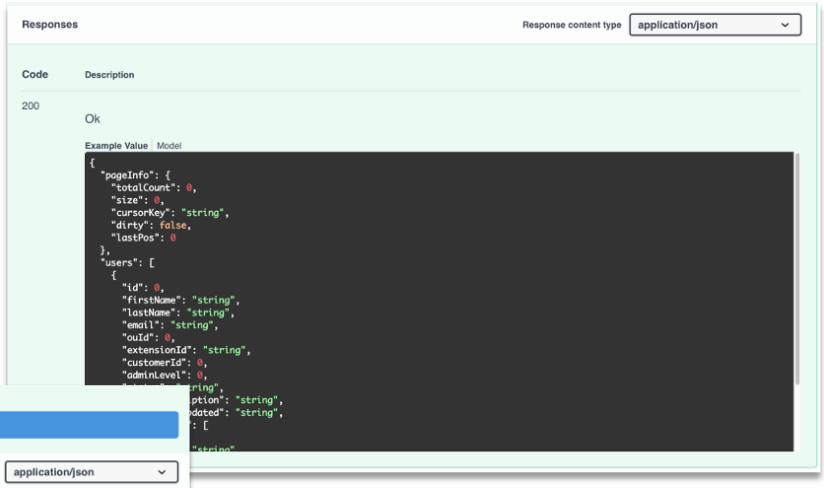
Click on an **API** to expand the contents — description, available request parameters and example Response Data — along with the **Try it out** button to make requests.

NOTE

*A **customerid** is not required to make API requests. However, one can still be included. If a customerId is included in the request, the Response Data will be based on the specified customerId. If there is no customerId in the request, the Response Data will be based on the customerId associated with the API Key being used. Specifying a customerId is primarily useful for partner or reseller accounts, that may want to use their own API Key to generate Response Data for one of their own customers.*

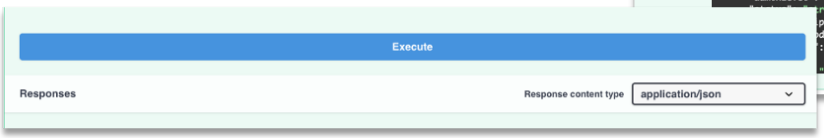


CommUnity



Each API will also include an example of the expected response body data.

Click the **Try it out** button to reveal the **Execute** button which will send the API request using the specified parameters.



A request with a successful response will include:

1. An example of the curl request that can be used to generate the result (with an icon to copy the request to the clipboard)
2. The request URL
3. The Server response code (I.e., 200)
4. The response body data (with icons to copy to clipboard and download)
5. The response headers

PAGINATION

Pagination is a method of breaking up large sets of data into smaller pieces. When making a request for data, you may not want to receive all of the data that is available in one response and instead may only want to work with a subset of the data at a time. When this is the case, Pagination breaks the data up into "pages" where each page contains a subset of the data.

APIs that support Pagination will have some type of limiter property (e.g., limit) which accepts an integer value of the number of records you want returned in each page. As an example, if you have a data set that contains 100 records, making a request with a limit of 10 will result in a response with the first page of 10 records.

Support for **Pagination** is only made available for certain APIs. You can determine if an API supports Pagination by checking the example of the Response Data. If the example response shows an object for "pageInfo" this is an indication that the API supports Pagination:

In addition to the limiter (e.g., limit) property, APIs that support Pagination also hav "offset" and "cursorKey" parameters that can be used in requests, to traverse Response Data across pages.

The offset parameter is an integer value that dictates the index of the last record received so that the next set of Response Data will start at the following index.

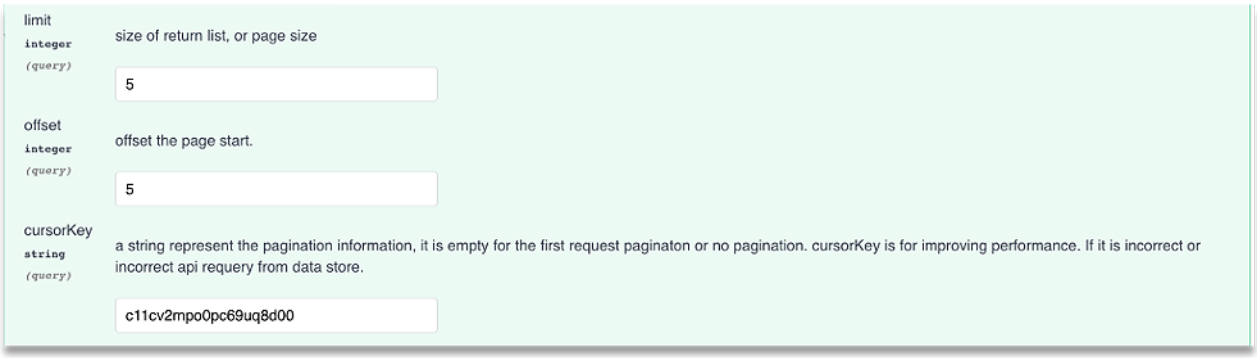
The cursorKey parameter is a string value used as a reference between Server and Database to more efficiently process paginated data. The cursorKey is returned in the response for paginated data requests and is not expected to be included as a parameter in the first request for paginated data. Using the cursorKey returned from the previous response in the next request made will be more performant by eliminating the need to requery the data store.

For example, the first request made for paginated data, includes a “limit” parameter with a value of 5.

The response data includes the pageInfo object, which includes the cursorKey and nextPos property:

{"pageInfo":{"totalCount":17,"size":5,"cursorKey":"c11cv2mpo0pc69uq8d00","dirty":false,"nextPos":5},

These properties can be used in the next fetch for data as the cursorKey and offset parameters respectively.

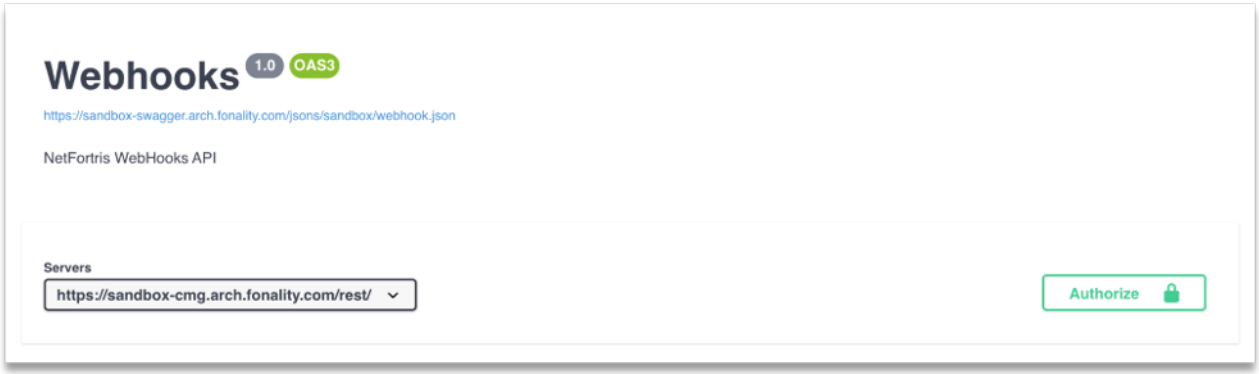


The next fetch of data will continue to increment based on the limit and offset specified in the request parameters:

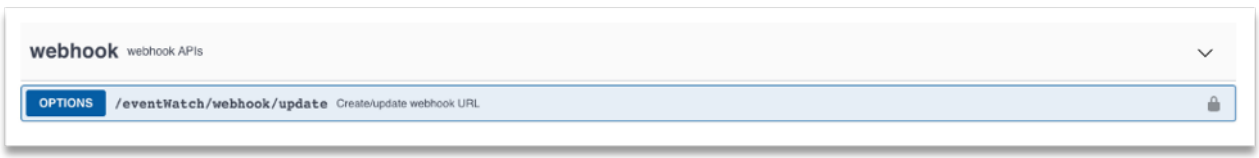
{"pageInfo":{"totalCount":17,"size":5,"cursorKey":"c11cv2mpo0pc69uq8d00","dirty":false,"nextPos":10},

WEBHOOKS

Webhooks currently operate in a “read only” state in this application and are only intended to be used as a reference for available Request Options and Response Data. The Header for Webhooks includes a Servers selector to switch between Servers. At this time, there is only one available Server, however, more may be added in the future.

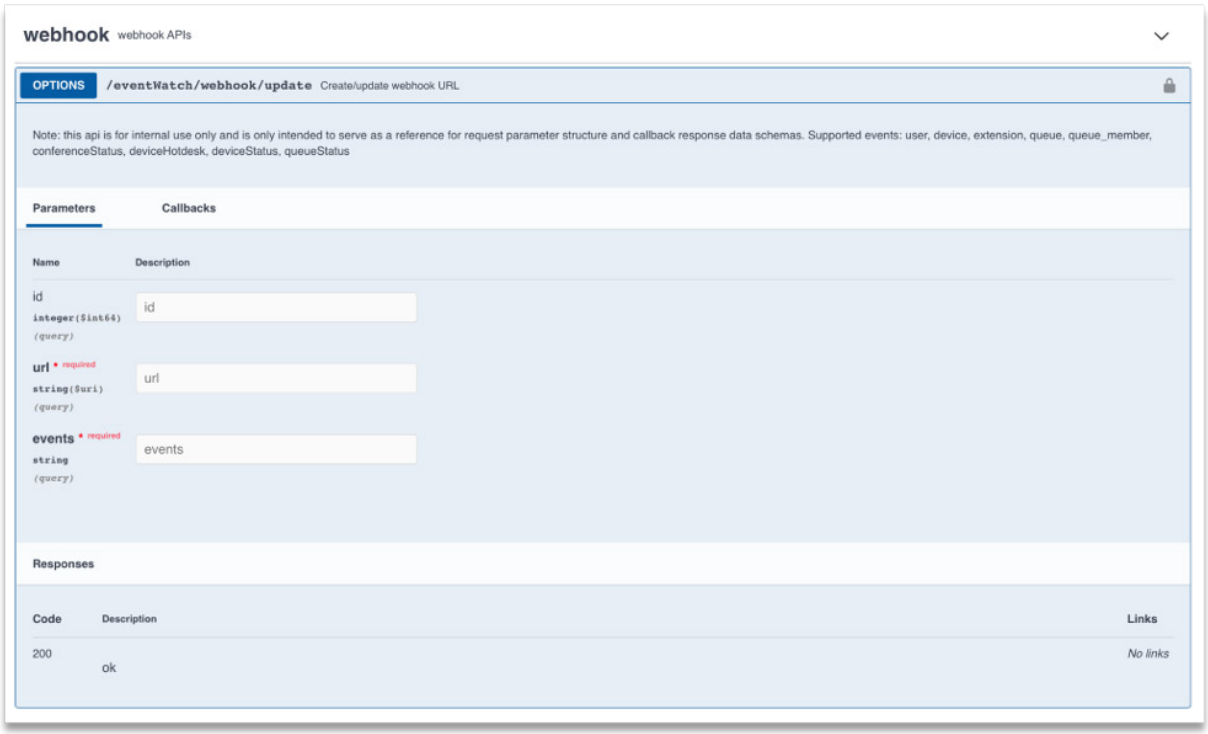


The available Webhooks are listed in this page. We currently only have one Webhook available.



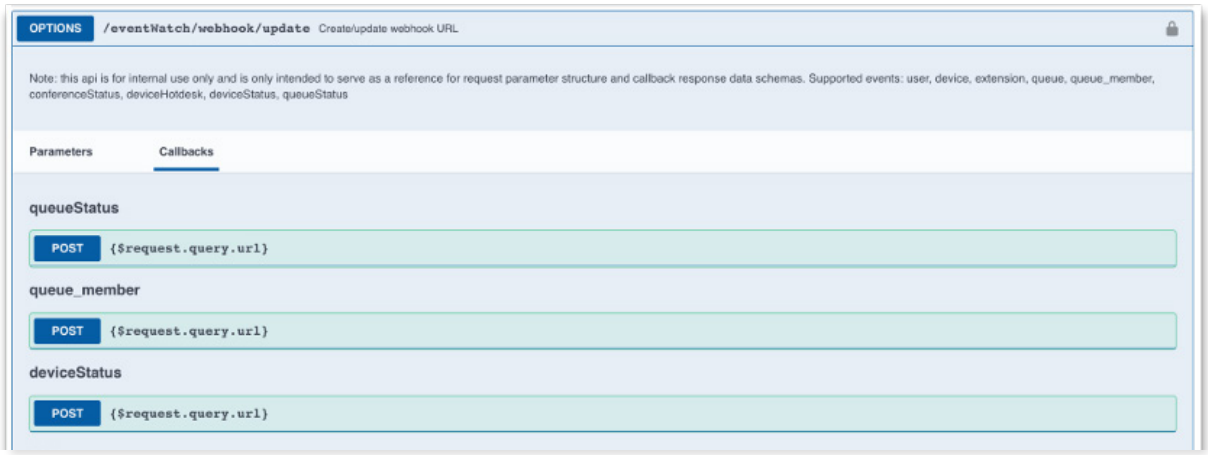
OPTIONS

Expanding the Webhook will reveal the options available for updating the Webhook (again, not available through this application), as well as available Callbacks. You will note that there is no **Try it out** button available.

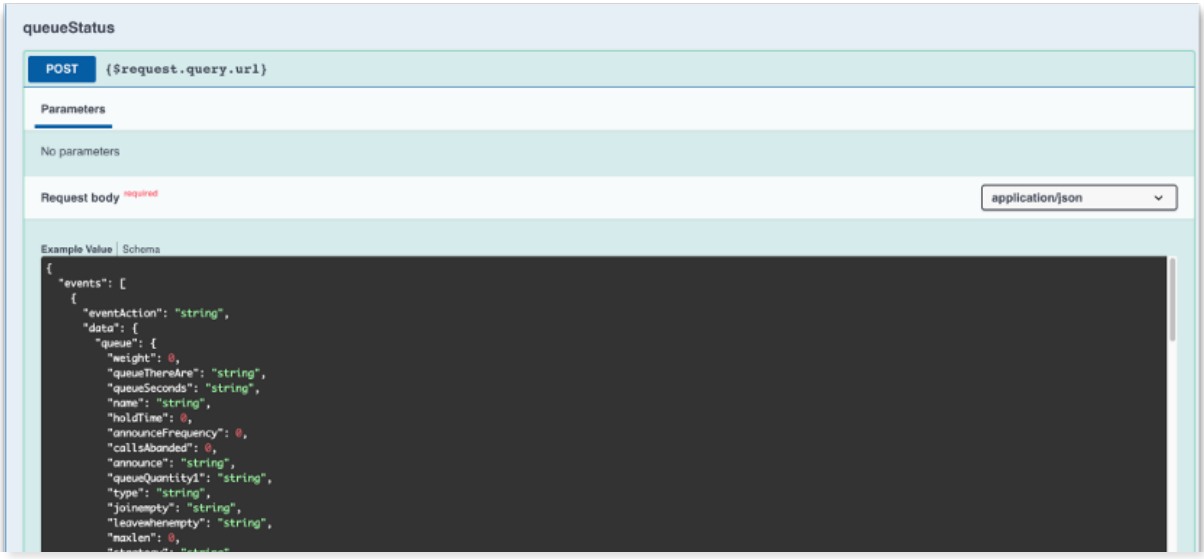


CALLBACKS

The Callbacks tab of the Webhook will list the available Callback functions that can be used to receive Response Data from the available Webhook event options. These Callbacks are again "read only".

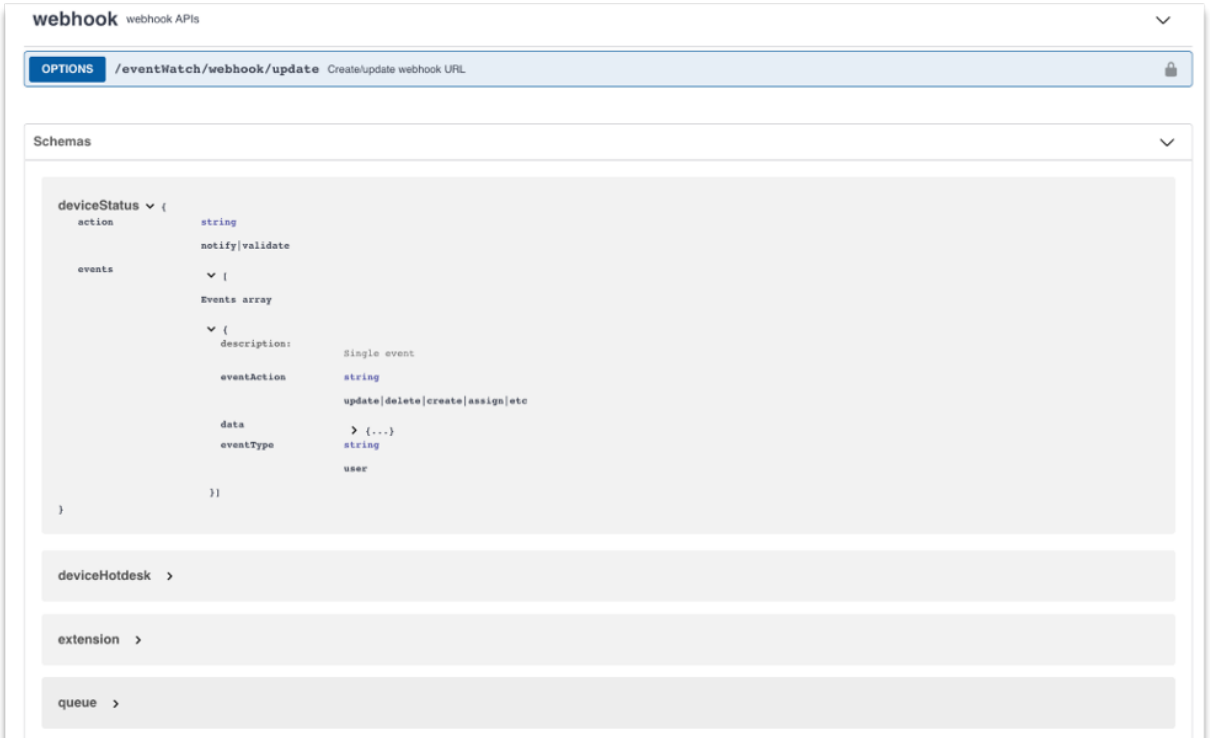


Expanding a Callback will reveal the structure of the event response data associated with the Callback.



As with APIs, Webhooks also includes the schema for Response Data objects related to Webhook events in the Footer.

Since we do not provide the ability to use Webhooks in this application, you will need to collaborate with a NetFortris representative to assist in the testing of Webhook Event Data.



TESTING WEBHOOKS

Important Note: You will need to have an API Key generated prior to generating a Webhook. If you do not yet have an API Key, please see the notes on generating an API Key towards the top of this document.

In order to generate a Webhook, you will need to reach out to a NetFortris representative and provide the following information:

- Your Customer ID, or Company Name
- The URL of your Server, for us to send Event Response Data to
- A list of events you would like to receive Response Data for

List of events:

- Queue Stats
- User
- Device Status
- Queue Member
- Conference Status
- Queue
- Device
- Extension
- Device Hotdesk

Once this information has been provided, a NetFortris representative will generate the Webhook for you.

PERMITTING WEBHOOK TRAFFIC

After the Webhook has been created, your Server is required to respond to a Validation Request with a Validation Code as a form of authorization, in order to receive Event Data. Once the Server has been validated, the Server can receive the event data related to the chosen Webhook events.

Please note that in order to receive the Validation Request and Event Data, you will need to make sure your network is configured to permit this traffic to reach your Server. If necessary, a NetFortris representative can provide you with the IP address that will be generating this data, so you can add rules to permit this traffic, along with providing assistance with the authorization process.

GENERATING WEBHOOK EVENT DATA

Once successfully authorized, your Server will receive event data, based on your Webhook configuration, whenever associated changes take place.

For example, if you chose to receive User Event Data, when making a change to a User (e.g., change the User's first or last name) in your Admin Panel, you will receive the event of the change on your Server.

Example Event Data

2021/03/04 00:50:32 Body {"action":"notify","eventType":"user","eventAction":"update","events":[{"prefix":"", "status":"active", "userId":12345,"customerId":1234,"extension":"107","ouId":123444,"firstName":"Joh","curExtensionId":1234,"email":"johndoe@netfortris.com","branchId":1234}]}

You can use the Webhook "Schemas" to view what kind of actions trigger Event Data as well as the data that is returned in each Webhook event.

If you have any questions regarding how to generate Event Data or issues receiving Event Data, please reach out to a NetFortris representative for further assistance.