Importing SSAPI from developers.calix.com to Postman

Owner
Trent Tate

Summary
In this video, we explore how to import API definitions from the Calix developer portal into Postman to streamline the process of working with these APIs. The guide assumes basic familiarity with Postman and the Calix developer portal. The tutorial covers setting up an API app, automating API call processes, handling OAuth 2.0 authentication challenges, and using Postman environment variables for efficient API prototyping. Learn how to import API tiles, clean up placeholder text, and manage your API keys and tokens effectively. This step-by-step guide ensures you can integrate and test Calix APIs with ease.

Community Link
community.calix.com/s/article/Importing-SSAPI-from-developers-calix-com-to-Postman (https://community.calix.com/s/article/Importing-SSAPI-from-developers-calix-com-to-Postman)

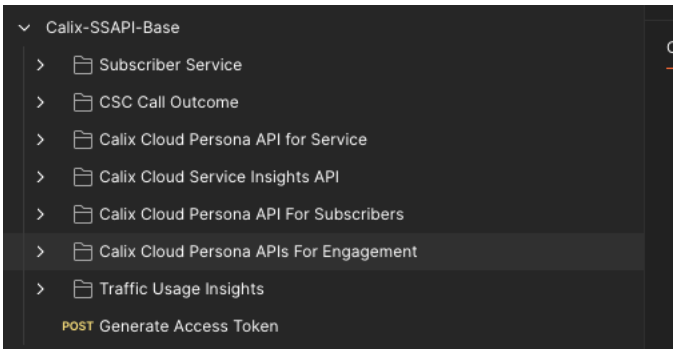
Content

Video

Background

Using the Calix Cloud APIs in the developers.calix.com (http://developers.calix.com/) portal is useful, but requires a lot of copy/paste of access tokens, and many developers may prefer to use tools like Postman as part of their day-to-day prototyping. We provide OpenAPI files to download, which can be imported into Postman, but there are some caveats.
Postman import does some specific things:

- Each imported call references a `{{baseUrl}}` variable, but does not populate anything from the `servers` directive in the OpenAPI spec into that variable.
  - In the Calix API set, the base URL will be the same for every call in the same YAML file. So we can set up a folder structure in Postman to automatically set the baseURL.
- This means for each API tile in developers.calix.com (http://developers.calix.com/), we need a folder in Postman, because each tile will contain multiple folders, and we only want to build our update script once. See below for more details on the scripts.

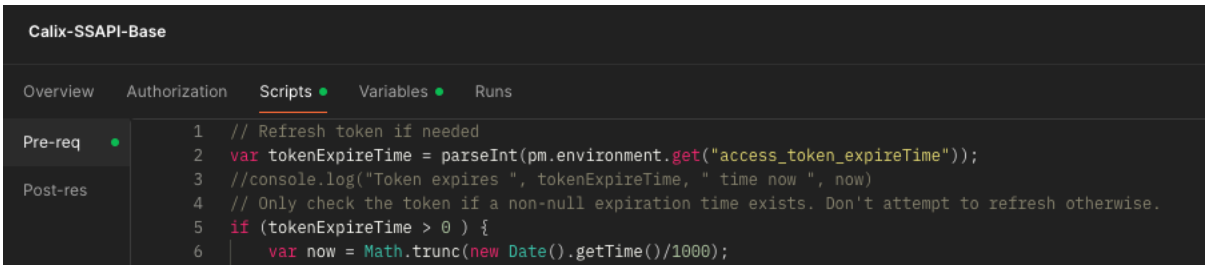


In addition, we need to care for the authorization piece, which Postman can't quite do on its own despite having support for a wide range of auth schemes.

Building your Own Postman collection

Start with template collection

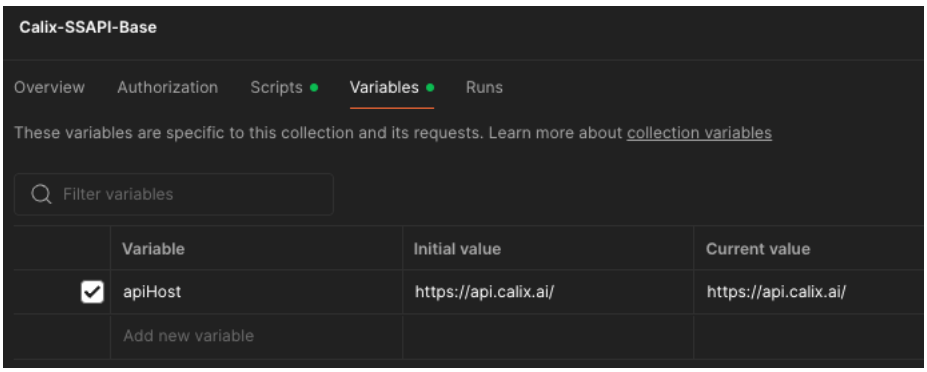
- Import the template collection into Postman (Calix-SSAPI-Base)
  Calix-SSAPI-Base-20240715.postman_collection.json (https://community.calix.com/s/article/Importing-SSAPI-from-developers-calix-com-to-Postman#) (attached to KB in Files section)
- This collection includes a pre-request script at the top level that handles token refresh:



```
// Only attempt to refresh the token if one exists - otherwise we need the initial grant via password in a separate call
// (maybe even the one we are currently processing!)
if (pm.environment.has("access_token")) {
    // If someone pasted in a token but no expire time, initialize it to 0
    if (! pm.environment.has("access_token_expireTime")) {
        pm.environment.set("access_token_expireTime", 0)
        console.log("Initialized access_token_expireTime = 0")
    }

    // Refresh token if needed
    var tokenExpireTime = parseInt(pm.environment.get("access_token_expireTime"));
    //console.log("Token expires ", tokenExpireTime, " time now ", now)
    var now = Math.trunc(new Date().getTime()/1000);
    if(now > (tokenExpireTime + 5)) {
        //console.log("--> token needs to be refreshed")
        pm.sendRequest({
            url: pm.variables.get("apiHost") + "/v1/authentication/token",
            method: 'POST',
            header: {
                'Accept': 'application/json',
                'Content-Type': 'application/x-www-form-rulencoded',
                'X-Calix-ClientID': pm.environment.get("client-id"),
            },
            body: {
                mode: 'urlencoded',
                urlencoded: [
                    {key: "grant_type", value: "refresh_token", disabled: false},
                    {key: "refresh_token", value: pm.environment.get("refresh_token"), disabled: false},
                    {key: "client_secret", value: pm.environment.get("client-secret"), disabled: false}
                ]
            }
        }, function(err, res) {
                pm.environment.set("access_token", res.json().access_token);
                if(res.json().expires_in) {
                    tokenExpireTime = parseInt(res.json().expires_in) + now;
                    console.log("New expiration time of ", now, " + ", parseInt(res.json().expires_in), " = ", tokenExpireTime)
                    pm.environment.set("access_token_expireTime", tokenExpireTime)
                }
            }
        );
    }
}
```

- It also includes the variable apiHost, used by the token refresh script. This should not change unless you are using a development environment or a non-US Cloud:



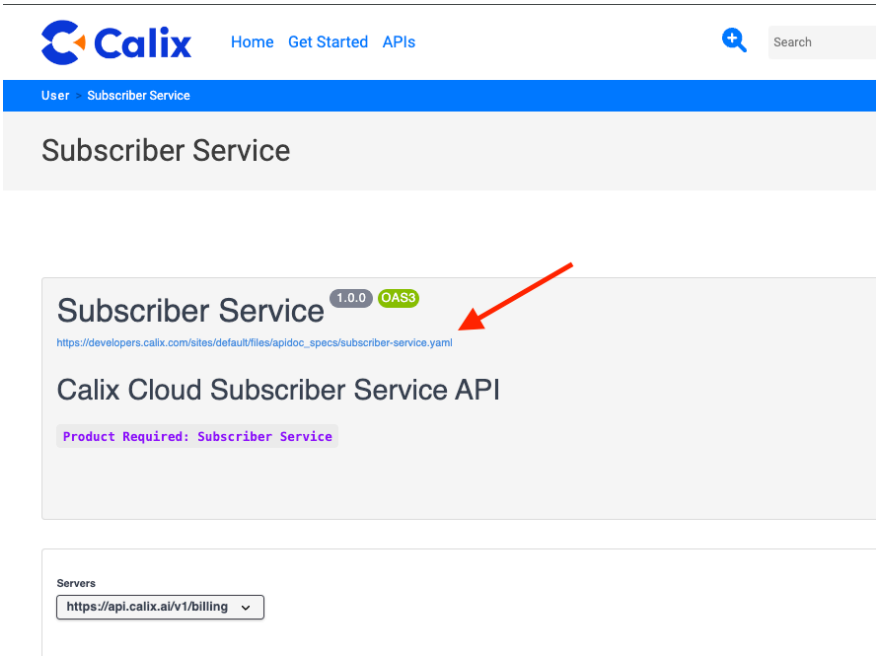
- As discussed above, it also includes a line of script at each folder level to set the base URL for that API tile. As more API tiles are published, you may have to duplicate one of these folders and update the script:

The Subscriber Services folder uses this line of script code:

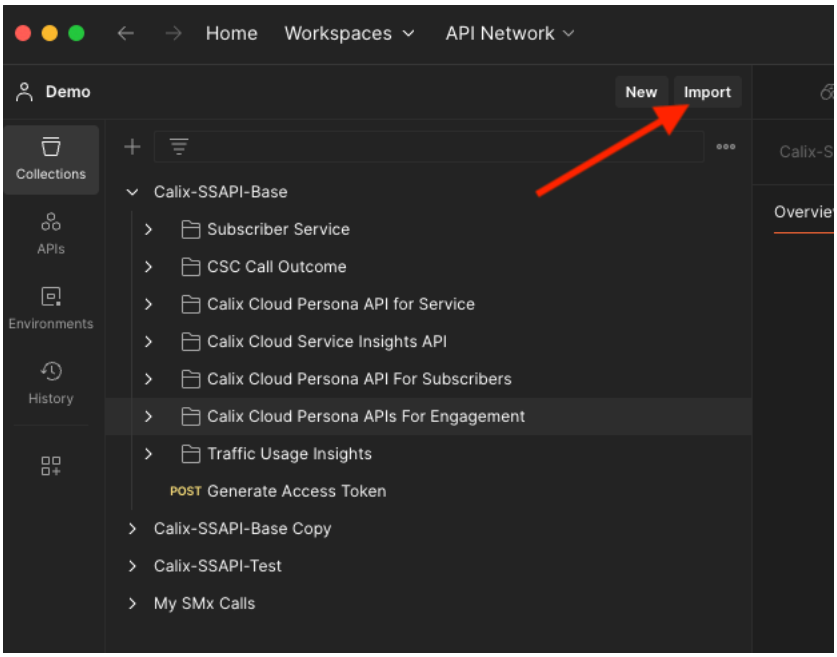
```
pm.environment.set("baseUrl", "https://api.calix.ai/v1/billing");
```

Export YAML from developers.calix.com and import to Postman

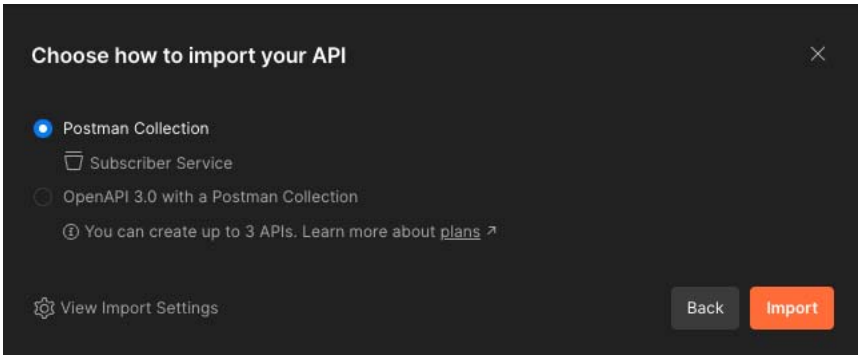
- Go in to each API tile you're interested in, and right-click to save the YAML:



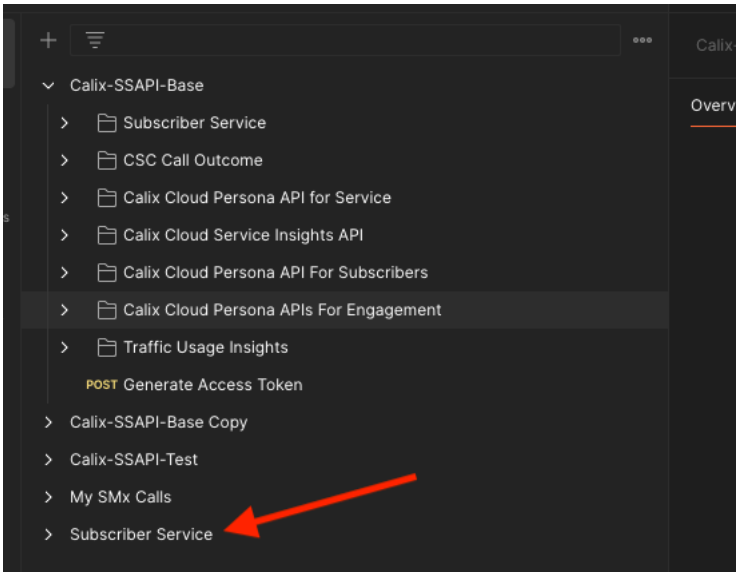
- Import the YAML into Postman:



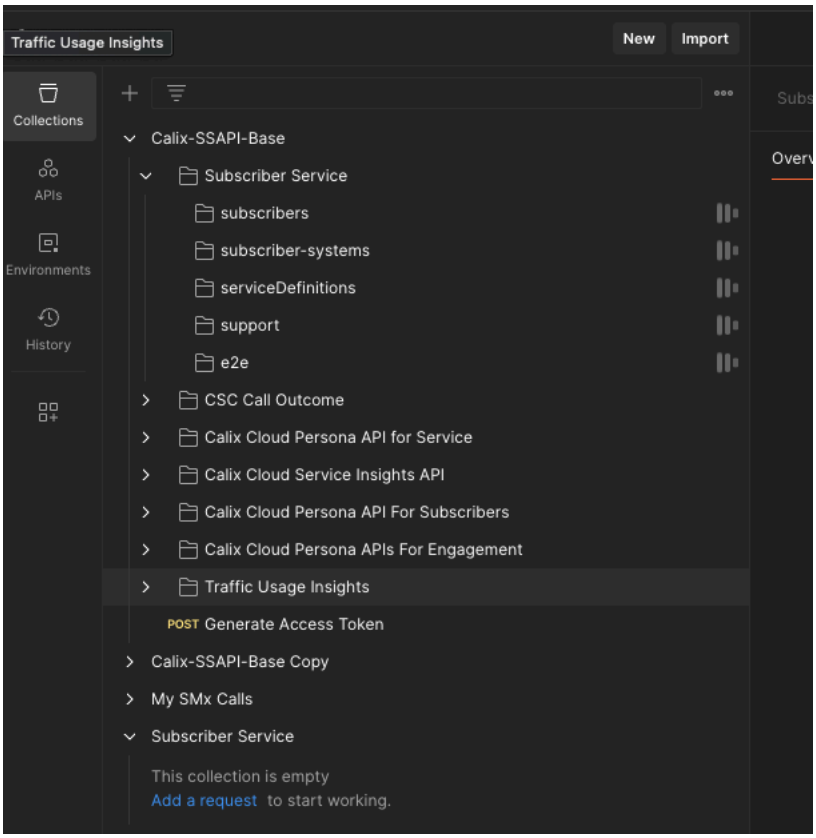
- Select the file you downloaded
- Import as Postman Collection



- This will import the API tile into a new collection:



- Open this collection, select all the folders in it, and drag them to your Calix-SSAPI-Base collection in the correct folder:

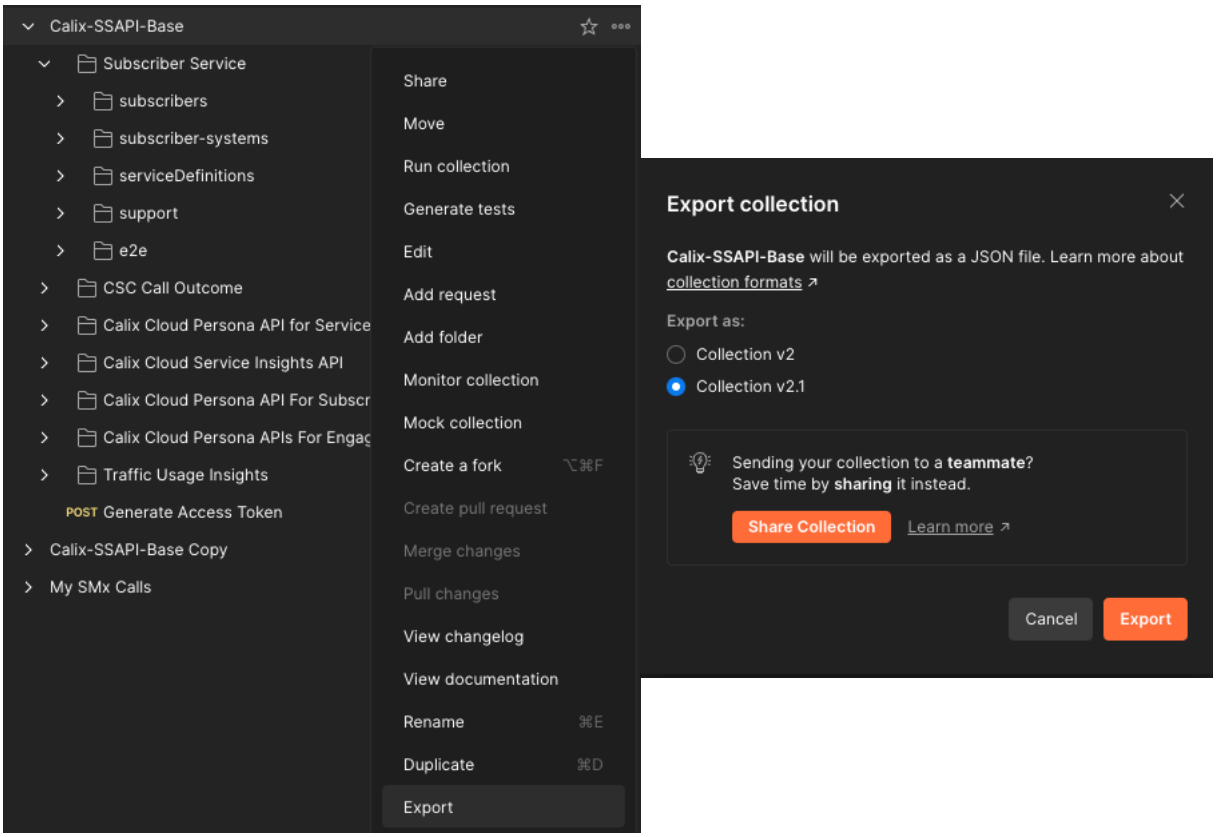


- Your imported collection will now be empty. Delete it.
- Repeat for each API tile you want to import.

Fix the Client-ID and Access Tokens in the collection

Every API call in the YAML file contains header values for the Client ID and Access Token. We want to replace those sample values with Postman variables. The best way to do this in bulk is to export the collection, edit it in a text editor, and import it back in to Postman.

- Right-click the collection, and Export:

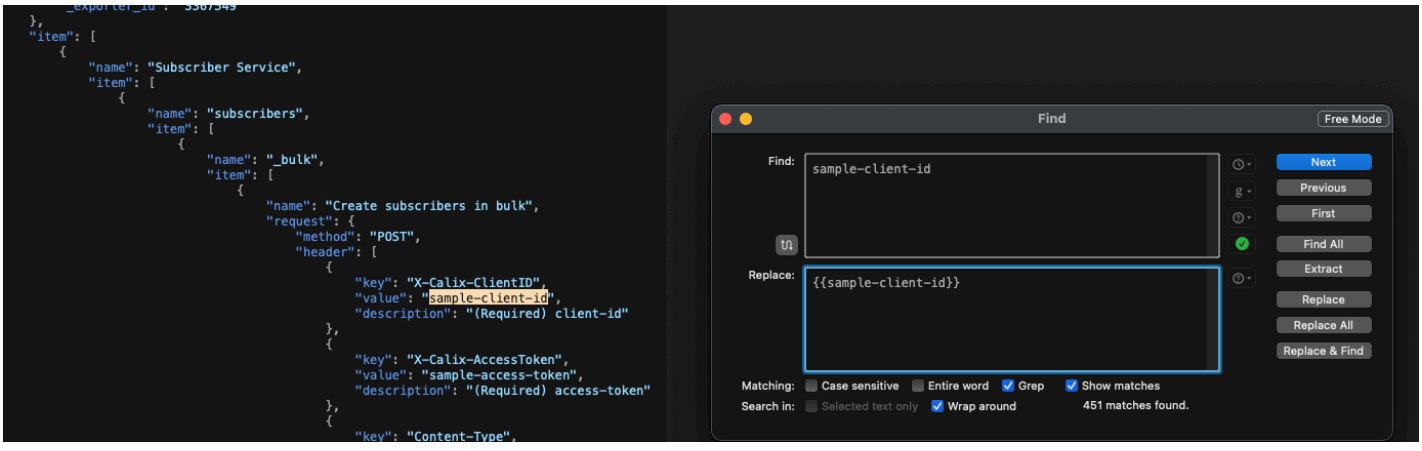


- Open the downloaded JSON file in a text editor
- Find and replace

```
sample-client-id
```

with

```
{{client-id}}
```



- Find and replace

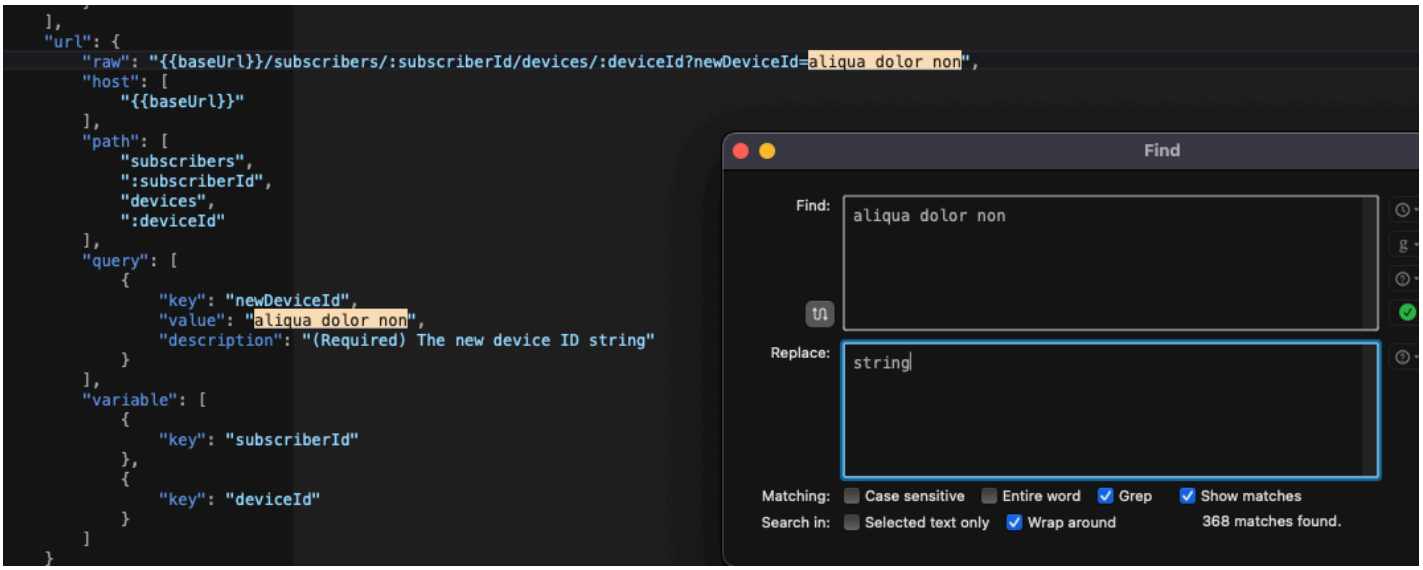
  ```
  sample-access-token
  ```

  with

  ```
  {{access_token}}
  ```

  (note the underscore)

- Postman will insert “lorem ipsum” type text for string value examples. The exact text changes each time you import YAML. If you want, find one of these strings and replace it with `string` or just empty text. You will have to replace a different string for each YAML file you imported earlier so there may be a few variants.



- Save your file in the text editor.

Import finished collection to Postman

- Import your edited collection. You can import it as a copy or replace your original.

Set up your Environment

You will need an Environment to store the variables used in this collection. Create a new Environment with these values:

Variable	Type	Initial value	(Note)
client-id	secret	Your client ID value	From developers.calix.com (http://developers.calix.com/) “Apps” tab
client-secret	secret	Your client secret	
baseUrl	default		
username	default	Your MyCalix username	
password	default	See note!	This is only used for the initial token grant; you may want to just substitute it for the single call rather than store it.

Variable	Type	Initial value	(Note)
These values will be created if they do not exist. If you have obtained an access/refresh token from the developers.calix.com (http://developers.calix.com/) UI, you may paste them here and will not need to use the username/password grant.			
refresh_token	default		
access_token	default		
access_token_expireTime	default	0	

You are now ready to make calls!

Last Published Date
7/19/2024, 09:40 AM

Post

Share an update...

Share

Sort by:

Most Recent Activity

Search this feed...

Collaborate here!

Here's where you start talking with your colleagues about this record.

Follow

Related Articles

Understanding Churn (/s/article/https-community-calix-com-s-group-0F94u000000Lp2kCAC-crush-the-competition) 94

Tools to view and help you write java/json files for use with the Consumer Connect Plus (CC+) API (/s/article/Tools-to-view-and-help-you-write-javajson-files-for-use-with-the-Consumer-Connect-Plus-CC-API-1) 390

Double Click on any of the Audit Log entries does not show details. (/s/article/Double-Click-on-any-of-the-Audit-Log-entries-does-not-show-details-1) 245

Engagement Cloud: 21.1 Release (/s/article/Calix-Cloud-UI-UX-Refresh-Marketing-Cloud-Homepage-and-Overview) 136

LAG between C7 and E7 bouncing with LAGDOWN alarm but no LOS on any ethernet port associated with the LAG on either side. (/s/article/LAG-between-C7-and-E7-bouncing-with-LAGDOWN-alarm-but-no-LOS-on-any-ethernet-port-associated-with-the-LAG-on-either-side-1) 212

Trending Articles

tacTV: EXOS Support User Default Credentials (20.3 Update)
(/s/article/TAC-TV-EXOS-Support-User-Default-Credentials-20-3-Update)

tacTV: How to configure the GigaSpire WAN for IPv6
(/s/article/TAC-TV-How-to-configure-the-GigaSpire-WAN-for-IPv6)

CMS Probe Overview and Troubleshooting
(/s/article/CMS-Probe-Overview-and-Troubleshooting)

tacTV: How to populate an AE ONT under an E7 EXA GE Port
(/s/article/TAC-TV-How-to-populate-an-AE-ONT-under-an-E7-EXA-GE-Port)

Success Best Practice: Configuring and Managing Wi-Fi Settings on GigaSpire Systems
(/s/article/Success-Best-Practice-Configuring-and-Managing-Wi-Fi-SSIDs-on-GigaSpire-BLAST-Systems)

Files (1)

Calix-SSAPI-Base.postman_collection
0.0 MB • json