Vertex Cloud Technical Documentation (For VTXCLOUD v4.0 and higher for Vertex v2)

Included Libraries

The VTXCLOUD4 library includes some dependent libraries that are used in the processes. The main pieces of these libraries are included in the VTXCLOUD4 library itself to make delivery easier. Those libraries are as follows:

- **GETURI** (yearly license fee) - This is a product from BVSTools.com that is used to make HTTP/S calls to web APIs, make RESTful requests, etc. The VTXCLOUD4 library has its own version installed directly in the VTXCLOUD4 library.
  NOTE: This version includes v12.00 or up for GETURI. In this version of GETURI the hashtags (#) are removed from the beginning of the GETURI function names. For example, #geturi_setValue() is now geturi_setValue().
- **YAJL** (open source) - This is a JSON generator/parser that is used for requesting and renewing OAuth 2.0 tokens. It may also be used in your own applications.
  https://www.scottklement.com/yajl/

Data Areas

- VCID - A numeric data area used to generate unique IDs for naming files, processes, etc.
- VERSION - The version of the VTXCLOUD4 library
- GU* - These are data areas used by GETURI.

Files

- **VCCTLPF - Control File**
  This file is used to store control values for processing. There are some values initially set up that should not be removed or modified. But, you can feel free to add any new

control values for your own applications and use the #vc_getControlValue() function to retrieve it's value.

The values for **CLIENTID** and **CLIENTSECRET** *should* be updated with information from your own Vertex account.

```
File Name . . . . VCCTLPF
  Library . . . .   VTXCLOUD
Format Descr  . .
Format Name . . . RVCCTL
File Type . . . . PF              Unique Keys - N

Field Name FMT Start Lngth Dec Key Field Description
VCKEY       A     1    20          Control Key
VCVALUE     A    21  1024          Control Value
```

- **VC001PF - Vertex Cloud Main Account File**
  This file is used to store Vertex Cloud users (client ids), tokens, refresh tokens and other information related to the client id.

  This is the file that is updated when a new client idis added or a token is refreshed. This file should not be altered manually unless you want to force a refresh of a token. In that case it is best to make the expiration date in the past so that on the next use of the account the token will be automatically refreshed.

```
File Name . . . . VC001PF
  Library . . . .   VTXCLOUD
Format Descr  . .
Format Name . . . RVCT001
File Type . . . . PF              Unique Keys - N

Field Name FMT Start Lngth Dec Key Field Description
VCUSER      A     1   256          User
VCACSTK     A   257  2048          Access Token
VCKTYPE     A  2305    32          Token Type
VCKEXP      Z  2337    26          Token Expire TimeStamp
VCTIME      Z  2363    26          Last Activity TimeStamp
```

Source Physical Files

- **QCOPYSRC** - This file contains all of the /copy members used in the Vertex Cloud applications as well as GETURI and YAJL.
- **QRPGLESRC** - This file contains the main processing programs as well as sample programs.

Binding Directories

The VTXCLOUD4 library contains a binding directory named VTXCLOUD. This binding directory is used when compiling programs and creating service programs.

This binding directory contains links to the Vertex Cloud functions as well as YAJL.

Subprocedures

The Vertex Cloud system includes a service program named F.VTXCLOUD. This service program contains some of the base functions used when working with the Vertex Cloud HTTP APIs.

The source physical file member VTXCLOUD4/QCOPYSRC member P.VTXCLOUD contains the prototype definitions and descriptions for available functions.

The main function that will be used in your applications will be the #vc_getToken() function, which is used to retrieve (and renew if expired) the OAuth 2.0 token used in HTTPS requests to the Vertex Cloud API library.

Control Values

The VTXCLOUD4 library contains control values used for processing in the VCCTLPF file. These are important for processing and should only be updated if any endpoints or URLs change.

They can also be updated when setting SANDBOX vs LIVE values.

- **CLIENTID** - This is the client ID for your application. You will need to update this before running any applications.
- **CLIENTSECRET** - This is the client secret for your application. You will need to update this before running any applications.
- **TOKENURL** - This is the URL to retrieve a token during the OAuth 2.0 process. (auth.vertexsmb.com/identity/connect/token)
- **APIBASEURL** - This is the base URL used when contacting the Vertex Cloud systems and their APIs. (restconnect.vertexsmb.com/vertex-restapi)
- **APISCOPE** - This is the scope used when requesting OAuth 2.0 tokens. (calc-rest-api)
- **GRANTTYPE** - This is the grant type used when requesting OAuth 2.0 tokens. (client_credentials)
- **DATAPATH** - This is the default path to where data files will be placed. (/vertexCloud/data)
- **DEBUGPATH** - This is the default path to where debug files will be placed. (/vertexCloud/debug)

- **OUTOUTPATH** - This is the default path to where output files from API call responses are palced. (/vertexCloud/output)
- **SALEURL** - This is the URL used (in conjunction with **APIBASEURL**) to make a request to the Sale API.
- **ADDRESSLOOKUPURL** - This is the URL used (in conjunction with **APIBASEURL**) for address lookups.
- **PURCHASEURL** - This is the URL used (in conjunction with **APIBASEURL**) for purchase orders.
- **DEBUG** - Turn debug on (*YES) or off (*NO).

Sample Programs

There are two sample programs in QRPGLESRC.

GETTOKEN is a sample program showing how to retrieve a token. This really won't be used much. It's just an example.

CALCTAX is a program that will make a request to calculate estimated tax. The JSON file containing the data we are sending to Vertex Cloud is in the IFS in /vertexCloud/data in a file named sale_sample.json. This is the exact file that Vertex Cloud uses in their example in the documentation.

PURCHASE is a sample program used to make a purchase order request.

TAXLOOKUP is a sample program used to retrieve tax information for an address.

These programs are used to show how to send JSON to the API. There is no JSON parsing in the program, but the data returned from the request will be stored in the IFS in /vertexCloud/output with a unique file name. The HTTP headers are also returned into their own separate file with the ".hdr" extension. There is a sample in the directory already.

As far as retrieving and refreshing tokens, that is all taken care of automatically in this case by the `#vc_getToken()` ILE function included in the F.VTXCLOUD service program.. You just need to be sure to supply the appropriate id and password in the control file.

The client ID and client secret are stored in a control file (VCCTLPF) as [described in the documentation](#).

If the client ID and/or client secret need to be different in some cases you can either set up new control values for a new set of client ID/Secret or hard-code them in the application.

Example as it is in the sample program:
```
clientID = #vc_getControlValue('CLIENTID');
```

```
clientSecret = #vc_getControlValue('CLIENTSECRET');
grantType = #vc_getControlValue('GRANTTYPE');

rc = #vc_getToken(clientID:clientSecret:grantType:token:errorMsg);
```

Example using different control values (the control values in this example would need to be set up in the VCCTLPF file):
```
clientID = #vc_getControlValue('NEWCLIENTID');
clientSecret = #vc_getControlValue('NEWCLIENTSECRET');

rc = #vc_getToken(clientID:clientSecret:grantType:token:errorMsg);
```

Example using static values:
```
clientID = 'STATICID';
clientSecret = 'STATICSECRET';

rc = #vc_getToken(clientID:clientSecret:grantType:token:errorMsg);
```