Exemplo Python - Como Criar uma Aplicação para Download dos Arquivos do Blob Storage Utilizando SAS Token

Esse documento é um exemplo de como fazer download dos arquivos no Blob Storage utilizando o SAS Token no Ambiente de Certificação.

Nesse exemplo foi utilizado:

Python 3.8

Bibliotecas PyPI:

- lxml (https://pypi.org/project/lxml)
- httpx (https://pypi.org/project/httpx)
- beautifulsoup4 (https://pypi.org/project/beautifulsoup4)

Abaixo está o código que pode ser usado para fazer download dos arquivos.

Configuração dos Imports

```
import asyncio
from http import HTTPStatus
from pathlib import Path
from urllib.parse import parse_qs, urlencode, urlparse, urljoin

from bs4 import BeautifulSoup
from httpx import AsyncClient
```

Variáveis

Variáveis

```
url_auth = 'https://api-listados-cert.b3.com.br/api/oauth/token'
sas_token_url = 'https://api-listados-cert.b3.com.br/api/cip/v1/sas-tokens'
grant_type = 'client_credentials'
client_id = '' # Incluir o client ID
client_secret = '' # Incluir o client secret
category = '' # Incluir a categoria do participante

# Caminho no disco dos arquivos do certificado digital
cert = (r'C:\PASTA\ARQUIVO.cer',
        r'C:\PASTA\ARQUIVO.key')

# Opcional. Apenas necessário se for usado um proxy
proxies = {
    'http://': 'http://IP:PORTA',
    'https://': 'https://IP:PORTA'
}
```

Código Utilizado para Fazer Download dos Arquivos

```
async def main():
        # Excluir o parâmetro proxies, se ele não for necessário
    async with AsyncClient(verify=True, cert=cert, proxies=proxies) as http_client:
        headers = {'CATEGORY_ID': category}

        data = {
            'client_id': client_id,
            'client_secret': client_secret,
            'grant_type': grant_type
        }

        response = await http_client.post(
            url_auth, data=data, headers=headers)

        if response.status_code == HTTPStatus.OK:
            access_token = response.json()['access_token']

            headers = {'Authorization': f'Bearer {access_token}'}

            response = await http_client.get(
                sas_token_url, headers=headers)

            if response.status_code == HTTPStatus.OK:
                blob_sas_token_url = response.json()['data']['sasToken']

                params = {
                    'comp': 'list',
                    'restype': 'container'
                }

                new_params = dict(parse_qs(urlparse(blob_sas_token_url).query))
                new_params.update(params)

                list_blob_url = urlparse(blob_sas_token_url)._replace(
                    query=urlencode(new_params, doseq=True)).geturl()

                response = await http_client.get(list_blob_url)

                blobs_xml = BeautifulSoup(response.text, 'xml')

                for blob in blobs_xml.Blobs:
                    if 'IMBARQ' not in blob.Name.text:
                        continue

                    file_name = blob.Name.text
                    download_url_path = urljoin(blob_sas_token_url, f'{urlparse(blob_sas_token_url).path}/
{file_name}')
                    download_url = f'{download_url_path}?{urlparse(blob_sas_token_url).query}'

                                        file_path = Path.cwd().joinpath(file_name)
                    await download_file(download_url, file_path)


async def download_file(url: str, file_path: Path):
    http_client = AsyncClient(proxies=proxies)

    with open(file_path, 'wb') as file:
        async with http_client.stream('GET', url) as http_response:
            async for data_chunk in http_response.aiter_bytes():
                file.write(data_chunk)


if __name__ == '__main__':
    asyncio.run(main())
```