AZURE BLOB / ADLS

AZURE BLOB:

Azure Blob Storage is a scalable, cloud-based object storage service from [Microsoft Azure](https://azure.microsoft.com/en-in/products/storage/blobs) for storing vast amounts of unstructured data, such as images, documents, video, audio files, and backups. It's ideal for various applications, including data lakes, mobile and web apps, high-performance computing, and machine learning, offering features like security, data management, and accessibility from anywhere via the internet.

How to access Azure blob using python:

Example using Azure SDK for Python:

Prerequisites:

Azure storage account with blob and container

Install python packages

access blob - pip install azure-storage-blob

password authentication - azure-identity

| Method | Description |
| --- | --- |
| upload\_blob(name, data) | Uploads a new blob |
| get\_blob\_client(name) | Returns a BlobClient |
| list\_blobs() | Lists blobs in the container |
| delete\_blob(name) | Deletes a blob |
| exists() | Checks if the container exists |
| get\_container\_properties() | Gets metadata and properties |

|  |  |
| --- | --- |
| get\_blob\_properties() | Fetches blob properties |

|  |  |
| --- | --- |
| set\_blob\_metadata() | Sets custom metadata |

|  |
| --- |
| open\_read() |

|  |
| --- |
| Opens the blob as a readable file-like object |

## 1. ****Access Public Blobs (No Authentication)****

If a blob container is **public**, you can access blobs using a simple HTTP request.

import requests

url = "https://<storage-account>.blob.core.windows.net/<container>/<blob-name>"

response = requests.get(url)

if response.status\_code == 200:

with open("downloaded\_file.txt", "wb") as f:

f.write(response.content)

## 2. ****Access Using SAS Token****

If the blob/container is **private**, use a **SAS token** to authenticate via URL:

import requests

url = "https://<storage-account>.blob.core.windows.net/<container>/<blob>?<SAS-token>"

response = requests.get(url)

if response.status\_code == 200:

with open("file.txt", "wb") as f:

f.write(response.content)

else:

print("Error:", response.status\_code, response.text)

## 3. ****Use REST API Directly (Advanced)****

You can interact with Azure Blob Storage via its REST API.

This requires:

* Constructing correct HTTP methods (GET, PUT, DELETE, etc.)
* Setting request headers like x-ms-date, x-ms-version
* Signing requests with the storage account key (HMAC-SHA256)

AZURE ADLS:

Azure Data Lake Storage (ADLS) is Microsoft's scalable, secure cloud storage service for big data analytics, combining the power of a [Hadoop-compatible file system](https://www.google.com/search?rlz=1C1GCEB_enIN1166IN1166&cs=0&sca_esv=f2b4c39e3966b02c&q=Hadoop-compatible+file+system&sa=X&ved=2ahUKEwjE-t2Gt62PAxW0hf0HHXvSLvwQxccNegQIAxAC&mstk=AUtExfDdWeo2PWMZBtf_s7Tzjo8kT1G3RFt1GxJ4MEX0NST9NjoUIKgU8XpNcd2eSY4uFIQxygzNkGWlSEJ4o5jWmymJ7fJ5YmLF-_Rm7cJe3nzku8NxskhY6x1NUhCPkwtfZnU&csui=3) with the economies of scale of [Azure Blob Storage](https://www.google.com/search?rlz=1C1GCEB_enIN1166IN1166&cs=0&sca_esv=f2b4c39e3966b02c&q=Azure+Blob+Storage&sa=X&ved=2ahUKEwjE-t2Gt62PAxW0hf0HHXvSLvwQxccNegQIAxAD&mstk=AUtExfDdWeo2PWMZBtf_s7Tzjo8kT1G3RFt1GxJ4MEX0NST9NjoUIKgU8XpNcd2eSY4uFIQxygzNkGWlSEJ4o5jWmymJ7fJ5YmLF-_Rm7cJe3nzku8NxskhY6x1NUhCPkwtfZnU&csui=3). It stores massive amounts of structured and unstructured data in its native format, allowing various analytics tools and engines to access and process it in a single repository for high-performance analysis.

ADLS Gen2 = Blob Storage + Hierarchical Namespace + Analytics Features

| Use Case | Recommendation |
| --- | --- |
| Store backups, media files, logs | **Blob Storage** |
| Work with big data, analytics, or machine learning tools | **ADLS Gen2** |
| Need directory-like structure & fine-grained access | **ADLS Gen2** |

How to access Azure ADLS uisng python:

using Azure SDK for python

1.same as blob

2.pip install azure-storage-file-data lake and authendication ( Storage account key  
• SAS token  
• Azure AD (service principal or managed identity))

from azure.identity import DefaultAzureCredential

from azure.storage.filedatalake import DataLakeServiceClient

# Replace with your storage account name

storage\_account\_name = "<your-storage-account-name>"

# Build the ADLS Gen2 endpoint URL

adls\_url = f"https://{storage\_account\_name}.dfs.core.windows.net"

# Authenticate using Azure AD

# DefaultAzureCredential works with:

# - Logged-in user via Azure CLI

# - Environment variables for service principal

# - Managed Identity (on Azure VMs, Functions, etc.)

credential = DefaultAzureCredential()

# Connect to Data Lake

service\_client = DataLakeServiceClient(account\_url=adls\_url, credential=credential)

# Access a specific container (file system)

file\_system\_name = "my-container"

file\_system\_client = service\_client.get\_file\_system\_client(file\_system\_name)

# List files and directories

print(f"Listing paths in '{file\_system\_name}':")

paths = file\_system\_client.get\_paths()

for path in paths:

print(" -", path.name)

without SDK:

### 1. ****Using a Shared Access Signature (SAS) token****

* Easiest and most common way.
* You generate a **SAS token** with the necessary permissions and include it in the URL.

### 2. ****Using Azure AD token + signed REST API****

* Much more complex (requires OAuth2, headers, canonical string signing).
* Use only if you **can’t** use a SAS token and want full control.

import requests

storage\_account = "mystorageaccount"

file\_system = "mycontainer"

file\_path = "folder/myfile.txt"

sas\_token = "<your-sas-token>" # Example: sv=2021-06-08&ss=b&srt=sco&sp=rl&se=2025-12-31T23:59:59Z&...

url = f"https://{storage\_account}.dfs.core.windows.net/{file\_system}/{file\_path}?{sas\_token}"

response = requests.get(url)

if response.status\_code == 200:

with open("downloaded\_file.txt", "wb") as f:

f.write(response.content)

print("File downloaded successfully.")

else:

print("Failed to download file:", response.status\_code, response.text)

import requests

tenant\_id = "<your-tenant-id>"

client\_id = "<your-client-id>"

client\_secret = "<your-client-secret>"

# OAuth 2.0 token endpoint for Azure AD

token\_url = f"https://login.microsoftonline.com/{tenant\_id}/oauth2/v2.0/token"

# Request parameters

data = {

"grant\_type": "client\_credentials",

"client\_id": client\_id,

"client\_secret": client\_secret,

"scope": "https://storage.azure.com/.default"

}

response = requests.post(token\_url, data=data)

token = response.json().get("access\_token")

if not token:

raise Exception("Failed to get access token")

print("Access Token:", token)

storage\_account = "<your-storage-account>"

file\_system = "<your-container-name>"

url = f"https://{storage\_account}.dfs.core.windows.net/{file\_system}?resource=filesystem"

headers = {

"Authorization": f"Bearer {token}",

"x-ms-version": "2021-06-08",

"x-ms-date": "Thu, 01 Jan 1970 00:00:00 GMT"

}

response = requests.get(url, headers=headers)

if response.status\_code == 200:

print("List of files/directories:")

print(response.text)

else:

print("Error:", response.status\_code, response.text)