Token credentials

What are they?

Token credentials originate from Microsoft's Azure core library as a way to encapsulate how a credential provides an OAuth token. A basic TokenCredential class will expose a `get_token` method that returns a valid access token and expiry date whenever it's called - hiding the fetching and refreshing logic from the calling code.

Example of a basic TokenCredential:

```
class TokenCredential(object):
  ...
  def get_token(self, *scopes, **kwargs):
    # obtain access token logic (fetch or refresh)
    access_token, expiry_time = fetching_logic()

    # AccessToken here is a named tuple
    return AccessToken(access_token, expiry_time)
```

These types of credentials are used extensively in Azure Python libraries as a way to provide *valid* OAuth tokens to different service client libraries (e.g. DataLakeServiceClient, EventHubConsumerClient, etc.).

You can also use these token credentials in your own code as it provides an easy way to continuously fetch a valid access token for a specific app.

Why do I need them in Hogwarts?

`hogwarts-auth`, the Hogwarts OAuth/OBO Python library, offers several TokenCredential classes to help simplify your code. In fact, these classes will take care of all interactions with the VaultClient (e.g. on-behalf-of, refreshing, etc.) and a few of them will even handle the *audience* and *scope* values so that you don't have to. Furthermore, these credentials are ready to use with the majority of Azure Python libraries.

Here are the token credentials Hogwarts currently offers:

TokenCredential (generic)

A generic token credential which requires the user to provide an *audience* and a *scope*.

```
import os
from hogwarts.auth.vault.token_credential import TokenCredential

datalake_cred = TokenCredential(audience="https://storage.azure.com", scope="user_impersonation")
```

App specific credentials

Azure apps :

- DataLakeTokenCredential
- ServiceBusTokenCredential**
- EventHubTokenCredential**

Hogwarts apps :

- AlfredTokenCredential**
- NbGalleryTokenCredential**
- TrinoTokenCredential**

** To be released soon

> In the upcoming release, credentials will be moved under `hogwarts.auth.vault.credentials`.

How do I use them in Hogwarts? (DataLake example)

> Hogwarts token credentials need to run inside of a Hogwarts execution platform (Airflow, JupyterHub) as it depends on the Hogwarts Vault for all OAuth operations.

First, install the Azure Datalake python library

```
! pip install azure-storage-file-datalake
```

Then, import the token credential from `hogwarts-auth` and instantiate it.

```
import os
from hogwarts.auth.vault.datalake_token_credential import DataLakeTokenCredential

# Works out-the-box on Airflow & JupyterHub
datalake_cred = DataLakeTokenCredential()
```

Create the service client (DataLakeServiceClient)

```
from azure.storage.filedatalake import DataLakeServiceClient

datalakename = "stpilotdatalakedev"

try:
    global service_client

    service_client = DataLakeServiceClient(account_url="{}://{}.dfs.core.windows.net".format(
        "https", datalakename), credential=datalake_cred)

except Exception as e:
    print(e)
```

Any interaction with the DataLake will now be authenticated as you

```
file_systems = service_client.list_file_systems()
for file_system in file_systems:
    print(file_system.name)

# Output
container1
```

```
container2
container3
```

As always, the Notebook code for this example can be found in the Hogwarts-example repository.