ps-plugin-MSGraphUtilities

MS Graph Utilities provides features to integrate with the Microsoft Graph API that cannot be achieved using an Appian Integration Object.

Connected Systems

OAuth 2.0 Client Credentials

Used to get access to the Graph API without a user via Application Permissions. See here for more details.

OAuth 2.0 Authorization Code

Used to get access to the Graph API on behalf of a user via Delegated Permissions. See here for more details.

Integrations

Large File Upload

This plugin provides 2 Integrations for large file upload via the Graph API, one for each type of Connected System. Each Large File Upload Integration provides the same functionality, but uses different authorization methods.

Enables large files to be uploaded via the Graph API. Files are uploaded in chunks of roughly 5MB. The standard Graph REST API for uploading a file is limited to files of up to 4MB.

> **For files under 4MB, it is recommended to use an HTTP Integration object to upload the file rather than using this plugin.**

On success, this Integration returns the following data in the Result object:

- id (Text) - the Drive Item Id for the uploaded document.
- eTag (Text) - the eTag for the uploaded document.
- name (Text) - the file name (including extension) of the uploaded document (which may differ from the Appian document's filename, e.g. when using Conflict Behavior "Rename").
- webUrl (Text) - the webUrl for the uploaded document.

Design Considerations

Scope

When using Authorization Code, add `offline_access` as a scope so that if the user's token has expired, Appian request a refreshed token without the user needing to reauthorize. It is not possible to use the `offline_access` scope alongside the Graph .default scope, so specific scopes need to be used, for example `https://graph.microsoft.com/Sites.ReadWrite.All offline_access`

This token refresh event will appear in the logs as follows:

```
2021-06-16 19:39:52 SEVERE [pool-21-thread-30]
com.microsoft.graph.logger.DefaultLogger.logError Throwable detail:
com.microsoft.graph.http.GraphServiceException: Error code:
InvalidAuthenticationToken
POST https://graph.microsoft.com/v1.0/drives/<DRIVE ID>/items/<ITEM ID>:/<FILE
NAME>:/microsoft.graph.createUploadSession
SdkVersion : graph-java/v3.7.0
```

Unsuccessful Requests

For most unsuccessful requests, the plugin will return an IntegrationError with the following properties:

- title : Error name (e.g. `Bad Request`)
- message : Json string containing the full exception returned by the Graph API. This field can be parsed to a dictionary using `a!fromJson` (though the value should be validated first as in the exception cases this field will not contain valid Json but rather an Appian-specified message). Having parsed this value, the exception fields are easily accessible (e.g. `responseCode`).
- detail : Response code and Error name (e.g. `400 Bad Request`)

Some exceptions to this are:

- when using the Authorization Code Connected System, Appian will return an IntegrationError with the message field populated with the text `No OAuth Access Token found for current user` if the user does not have an Access Token cached. On receiving this response, the user should be prompted to authorize as documented in https://docs.appian.com/suite/help/latest/Oauth_connected_system.html#providing-users-a-way-to-authorize.
- when required fields are missing, Appian will return an IntegrationError such as:
  - title : `Missing Required Fields`
  - message : `The following values are null or empty: targetFolderItemId`
  - detail : `Check that expressionable fields are not evaluating to null`

Expected Behavior

- Using Conflict Behavior "Fail" when a document of the same name exists will return **409 Conflict**.
- Using Conflict Behavior "Replace" when providing an eTag that does not represent the current version of that document will return **412 Precondition Failed**. When no eTag is provided, no optimistic locking validation will occur.
- An eTag will only be sent (in an *if-match* header) if a Target Document Drive Item Id is also provided. If Conflict Behavior "Replace" is used and no Target Document Drive Item Id is provided, the eTag will not be sent and the update will happen without validating the eTag.

- If a Target Document Drive Item Id is provided but Conflict Behavior is set to "Rename", a new document will be created rather than updating the targeted document. Use "Replace" instead to update the existing document.
- An invalid Drive Id will return **400 Bad Request**
- An invalid Target Folder Item Id will return **404 Not Found**
- An invalid Tenant Id, Client Id or Client Secret will result in a failure to authorize the Connected System:
    - `AADSTS90002: Tenant '<TENANT ID>' not found`
    - `AADSTS700016: Application with identifier '<CLIENT ID>' was not found`
    - `AADSTS7000215: Invalid client secret is provided`