Microsoft Graph API Provider Setup

Last Modified on 12/30/2019 8:47 pm EST

To authenticate a Microsoft Graph connector instance you must register an app with Microsoft. Then when you authenticate, use the **Application Id**, **Password/PublicKey**, and **Redirect URL** from your registered app as the **API Key**, **API Secret**, and **Callback URL**. If you plan to monitor events, also configure the **Webhook URL**.

If you've already set up an app and just need to know how to find your **Application Id** and **Password/PublicKey**, see Locate Credentials for Authentication. If you need to register an app, see Create an Application.

See the latest setup instructions in the Microsoft documentation.

On this page

Locate Credentials for Authentication

If you already created an application, follow the steps below to locate the **Application Id**, **Password/PublicKey**, and **Redirect URL**. If you have not created an app, see Create an Application.

To find your OAuth 2.0 credentials:

1. Log in to your account at Microsoft.
2. Click the application that you want to connect.
3. Record the **Application Id**.
4. If you don't know the **Password/PublicKey**, click **Generate New Password** to get a new one and record it.
5. Record the **Redirect URL** for your app.

Create an Application

If you have not created an application, you need one to authenticate with Microsoft.

To create an application:

1. Log in to your account at Microsoft.
2. Click **Add an App**.
3. Enter a name, and then click **Create**
4. Record the **Application Id**.

5. Under **Application Secrets**, click **Generate New Password**, record the **Password/PublicKey**, and then click **OK**.

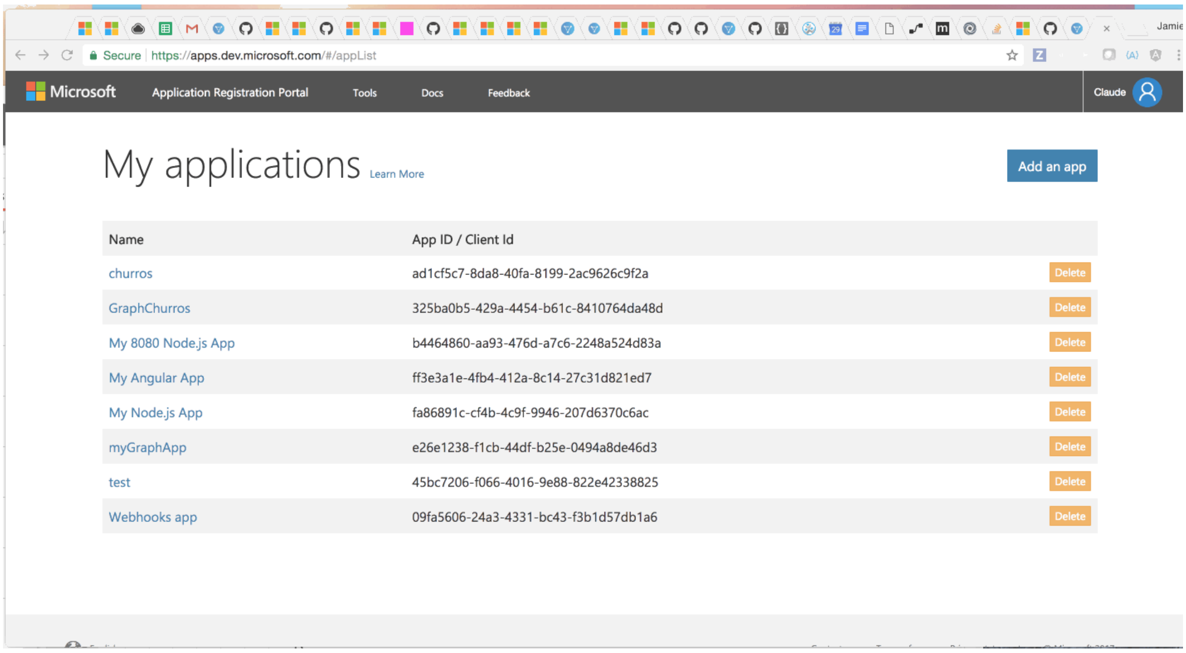
   > ⚠ **Important:** You cannot show the Password/Public Key again, so you will need to generate a new one if it's lost.

6. Under **Platforms**, click **Add Platform**, and then select **Web**.

7. In **Redirect URLs** enter the URL to redirect the user to at the end of the OAuth 2.0 authentication process. For example, the SAP Cloud Platform Open Connectors 2.0 callback URL is `https://auth.cloudelements.io/oauth`.

8. Under **Microsoft Graph Permissions** add the permissions needed to use your app. See Permissions for details.

9. Save your app.


Secure
https://apps.dev.microsoft.com/#/appList
Microsoft
Application Registration Portal
Tools
Docs
Feedback
Claude

My applications Learn More

Add an app

Name	App ID / Client Id	
churros	ad1cf5c7-8da8-40fa-8199-2ac9626c9f2a	Delete
GraphChurros	325ba0b5-429a-4454-b61c-8410764da48d	Delete
My 8080 Node.js App	b4464860-aa93-476d-a7c6-2248a524d83a	Delete
My Angular App	ff3e3a1e-4fb4-412a-8c14-27c31d821ed7	Delete
My Node.js App	fa86891c-cf4b-4c9f-9946-207d6370c6ac	Delete
myGraphApp	e26e1238-f1cb-44df-b25e-0494a8de46d3	Delete
test	45bc7206-f066-4016-9e88-822e42338825	Delete
Webhooks app	09fa5606-24a3-4331-bc43-f3b1d57db1a6	Delete

Permissions

When creating your app be aware of the permissions that you set. Each resource requires specific permissions. For example, the Calendar resource requires:

Permission Type	Permissions

Permission Type	Permissions
Delegated (work or school account)	Calendars.Read, Calendars.ReadWrite
Delegated (personal Microsoft account)	Calendars.Read, Calendars.ReadWrite
Application	Calendars.Read, Calendars.ReadWrite

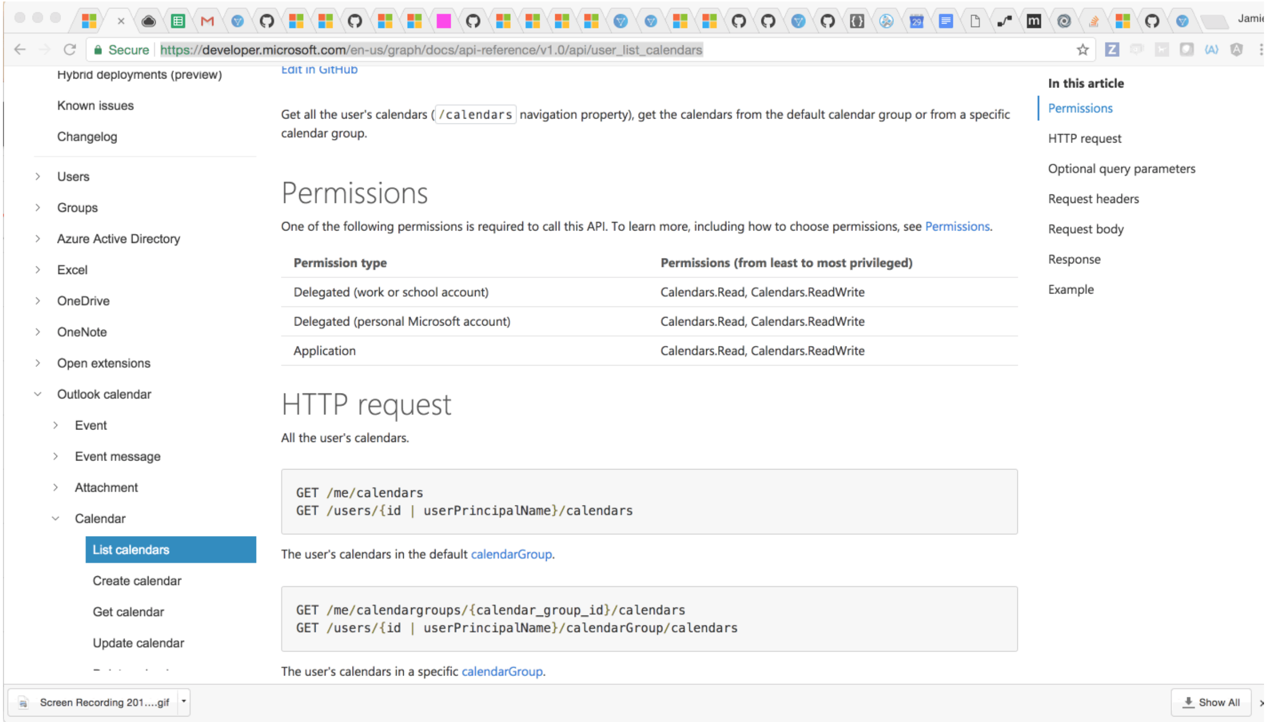
For more information see: https://developer.microsoft.com/en-us/graph/docs/api-reference/v1.0/api/user_list_calendars.

When setting up your app, in the Delegated Permission section you need:

- Calendars.Read
- Calendars.ReadWrite

In addition, to authenticate you need:

- Delegated Permission: Users.Read.All (if an admin Users.Read)
- Delegated Permission: Users.ReadWrite.All (non-admin Users.ReadWrite)
- Application Permissions: Users.Read.All
- Application Permissions: Users.ReadWrite.All.



Edit in GitHub

Get all the user's calendars (`/calendars` navigation property), get the calendars from the default calendar group or from a specific calendar group.

Permissions

One of the following permissions is required to call this API. To learn more, including how to choose permissions, see Permissions.

Permission type	Permissions (from least to most privileged)
Delegated (work or school account)	Calendars.Read, Calendars.ReadWrite
Delegated (personal Microsoft account)	Calendars.Read, Calendars.ReadWrite
Application	Calendars.Read, Calendars.ReadWrite

HTTP request

All the user's calendars.

```
GET /me/calendars
GET /users/{id | userPrincipalName}/calendars
```

The user's calendars in the default calendarGroup.

```
GET /me/calendargroups/{calendar_group_id}/calendars
GET /users/{id | userPrincipalName}/calendarGroup/calendars
```

The user's calendars in a specific calendarGroup.