



# Adding optional claims and custom scope in Access token

Version 0.1 (Aug-2024)



# Contents

- Overview ..... 3
- Prerequisite ..... 3
- Configure Optional Claims..... 3
- Configure Custom Scope ..... 5
- Configure Custom Scopes in API Permissions ..... 8
- Code Snippet for Acquiring the Access token with Custom Scope ..... 9

## Overview

This document outlines the steps for configuring optional claims and custom scopes in an Azure AD App and demonstrates how to use the AppManager API/libraries to acquire an access token with optional claims and custom scope.

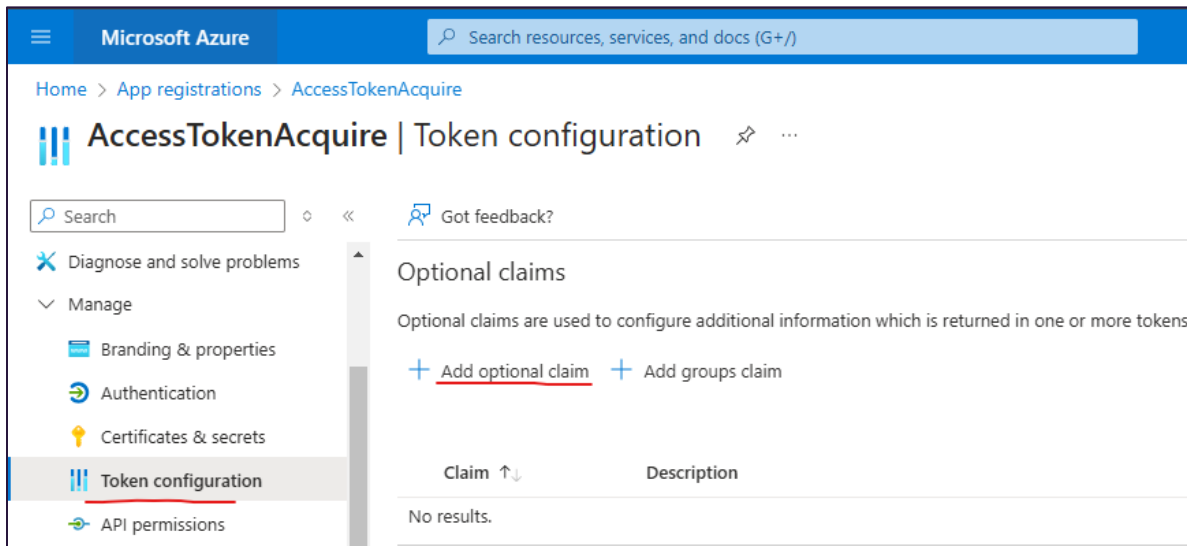
## Prerequisite

- To configure an Azure AD app, you will need the following prerequisites:
- **Azure Subscription:** An active Azure subscription.
- **Azure AD Tenant:** Access to an Azure Active Directory tenant.
- **Azure AD Admin Permissions:** Administrator privileges in the Azure AD tenant.
- **Azure Portal Access:** Access to the Azure portal.
- **Registered Application:** An application registered in Azure AD.
- **App Registration Permissions:** Appropriate permissions to configure settings on the registered application.

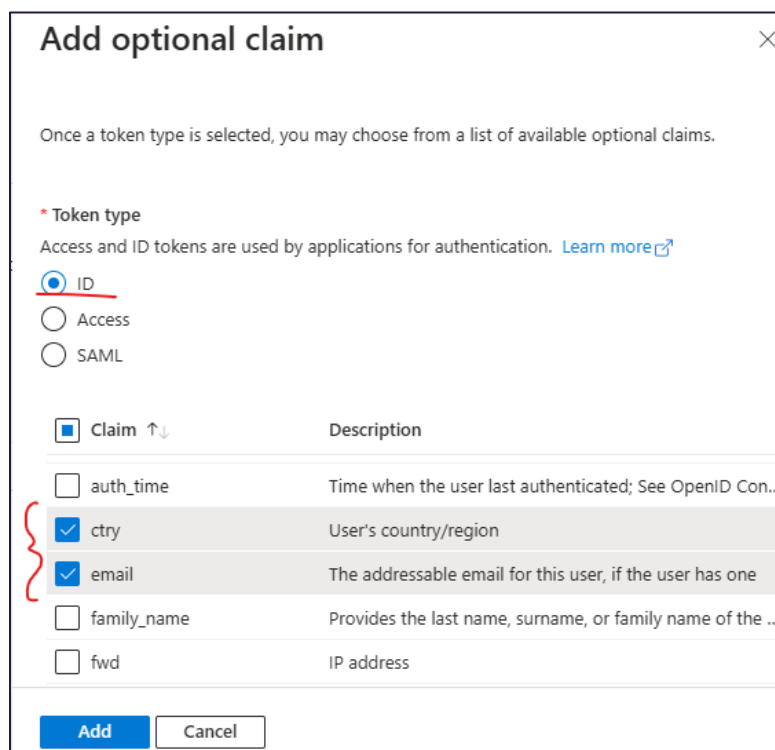
## Configure Optional Claims

You can configure optional claims for your application through the Microsoft Entra admin center's applications UI by following below steps:

1. Sign in to the Azure Portal and navigate to App registrations
2. Choose the application for which you want to configure optional claims based on your scenario and desired outcome.
3. Under Manage, select Token configuration.
4. Select Add optional claim.



5. Select the token type you want to configure, such as Access.
6. Select the optional claims to add.



7. Select Add. Once you add all the optional claims you should see all your claims as shown below:

**AccessTokenAcquire | Token configuration**

Search  Got feedback?

Diagnose and solve problems

Manage

- Branding & properties
- Authentication
- Certificates & secrets
- Token configuration**
- API permissions
- Expose an API
- App roles
- Owners

Optional claims

Optional claims are used to configure additional information which is returned in one or more tokens. [Learn more](#)

+ Add optional claim + Add groups claim

Claim ↑↓	Description	Token type ↑↓	Optional settings
ctry	User's country/region	ID	-
ctry	User's country/region	Access	-
email	The addressable email for this user, if the user has one	ID	-
email	The addressable email for this user, if the user has one	Access	-

## Configure Custom Scope

1. Under Manage, select Expose API.

Note: You may rename the Application ID URI to format: `api://{app's client Id}`

Home > App registrations > AccessTokenAcquire

**AccessTokenAcquire | Expose an API**

Search  Got feedback?

Diagnose and solve problems

Manage

- Branding & properties
- Authentication
- Certificates & secrets
- Token configuration
- API permissions
- Expose an API**
- App roles
- Owners
- Roles and administrators

Got a second to give us some feedback? →

Application ID URI :  Edit

Scopes defined by this API

Define custom scopes to restrict access to data and functionality protected by the API. An application that requires access to parts of this API can request that a user or admin consent to one or more of these.

Adding a scope here creates only delegated permissions. If you are looking to create application-only scopes, use 'App roles' and define app roles assignable to application type. [Go to App roles](#).

+ Add a scope

Scopes	Who can consent	Admin consent display ...	User consent display na...	State
api://[redacted] user_i...	Admins and users	Access AccessTokenAcquire	Access AccessTokenAcquire	Enabled

2. Click on Add a scope

enAcquire

Expose an API

Got feedback?

Application ID URI :

Scopes defined by this API

Define custom scopes to restrict access to data and functionality protected by the API. API can request that a user or admin consent to one or more of these.

Adding a scope here creates only delegated permissions. If you are looking to create a type. [Go to App roles.](#)

+ Add a scope

Scopes	Who can consent
api://[redacted]	Admins and users

Authorized client applications

Authorizing a client application indicates that this API trusts the application and users of this API.

### Add a scope

Scope name \* ?

api://[redacted]/

Who can consent? ?

☐ Admins and users ☒ Admins only

Admin consent display name \* ?

Admin consent description \* ?

User consent display name ?

User consent description ?

### 3. Add User.Read scope

This scope is essential to complete this configuration; without it, you will not be able to acquire the access or ID tokens with the custom scope.

### Add a scope

Scope name \* ?

✓

api://[redacted]/User.Read

Who can consent? ?

☒ Admins and users ☐ Admins only

Admin consent display name \* ?

✓




Admin consent description \* ?

User consent display name ?

User consent description ?

4. Add other scope based on your security requirements, for example adding another scope for People Search

### Add a scope ✕

 Save  Discard  Delete

Scope name \* ⓘ

People.Search

api://[redacted]/People.Search

Who can consent? ⓘ

Admins and users

Admins only

Admin consent display name \* ⓘ

Search People ✓

Admin consent description \* ⓘ

Allows app to search people

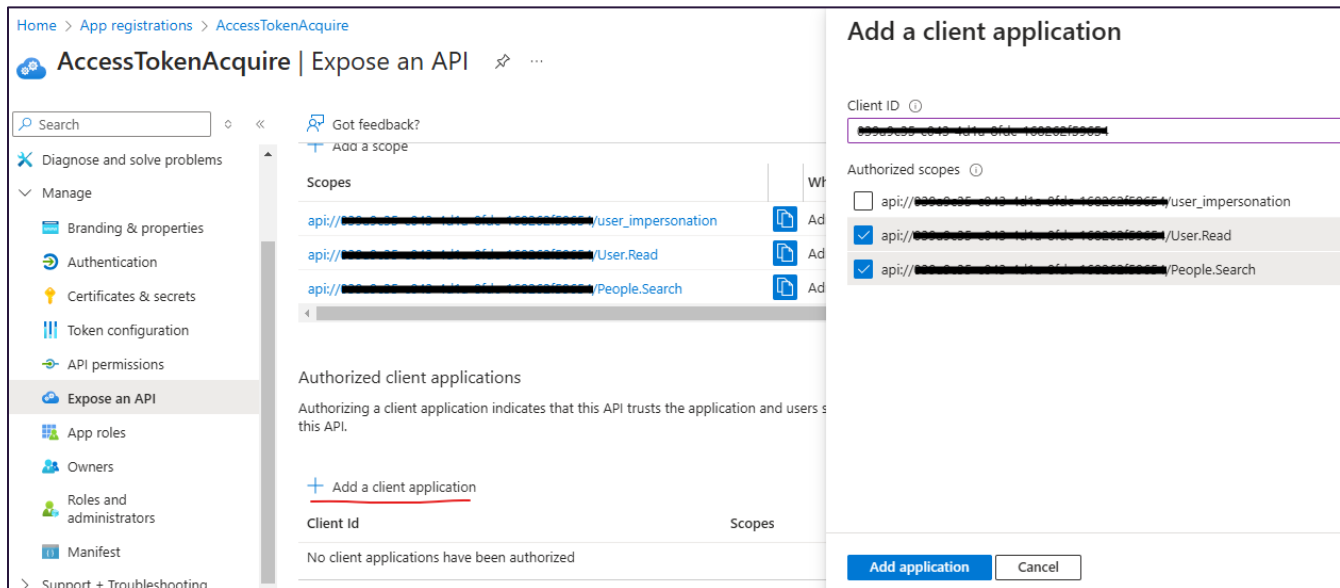
User consent display name ⓘ

e.g. Read your files

User consent description ⓘ

e.g. Allow the app to read your files

5. Add a client application

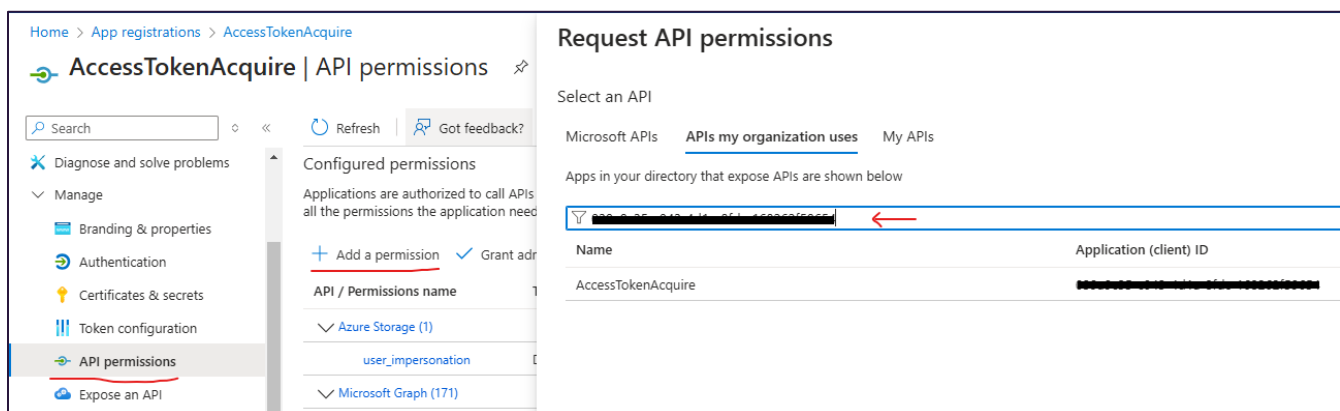


6. Enter client ID - should you're apps client/application Id and select the boxes against the scope that you want to add to token. Click on App application button.

## Configure Custom Scopes in API Permissions

You can configure custom scope in API permissions for your application through the Microsoft Entra admin center's applications UI by following below steps:

1. Under Manage, select API Permissions > Add Permissions > APIs my organization uses and search with the apps client/application Id that you have added in previous step



2. Select the client/application Id from search result and select the User.Read (must have) and other required permissions, for example - People.Search and click on Add Permissions.



Select permissions

Start typing a permission to filter these results

Permission	Admin consent required
> Other permissions	
<div> People (1) </div>	
<input checked="" type="checkbox"/> <div> People.Search ⓘ  Serch People </div>	No
<div> User (1) </div>	
<input checked="" type="checkbox"/> <div> User.Read ⓘ  Read Users </div>	No

Add permissions
Discard

3. Click "Grant admin consent for ..."

Configured permissions

Applications are authorized to call APIs when they are granted permissions by users/admins as part of the consent process. The list of configured permissions should include all the permissions the application needs. [Learn more about permissions and consent](#)

+ Add a permission
Grant admin consent for Akumina Inc.,

API / Permissions name	Type	Description	Admin consent requ...	Status
<div> AccessTokenAcquire (2) </div>				
People.Search	Delegated	Search People	No	...
User.Read	Delegated	Read Users	No	...

## Code Snippet for Acquiring the Access token with Custom Scope

Code

```
var customScopeResponse = await client.QueryAccessToken(Request.QueryString,
refreshTokenFromAccessCode, "api://xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxxx", "");
```

Where **xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxxx** is your app's client Id

Access token from above code, contains optional claims - **ctry** and **email** and custom scopes - **User.Read** and **People.Search**

```
https://jwt.ms

jwt.ms

{
  "appid": "039a9c35-0313-4d1a-8fde-168262f59854",
  "appidacr": "1",
  "ctry": "US",
  "email": "luke.shuck@akuminadev02.onmicrosoft.com",
  "family_name": "Shuck",
  "given_name": "Luke",
  "ipaddr": "10.10.10.10",
  "name": "Luke Shuck",
  "oid": "00000000-0000-0000-0000-000000000000",
  "rh": "00000000-0000-0000-0000-000000000000",
  "scp": "People.Search.User.Read",
  "sub": "00000000-0000-0000-0000-000000000000",
  "tid": "00000000-0000-0000-0000-000000000000",
  "unique_name": "Luke.Shuck@akuminadev02.onmicrosoft.com",
  "upn": "Luke.Shuck@akuminadev02.onmicrosoft.com",
  "uti": "00000000-0000-0000-0000-000000000000",
  "ver": "1.0"
}.[Signature]
```