Introduction to the Microsoft Identity Platform

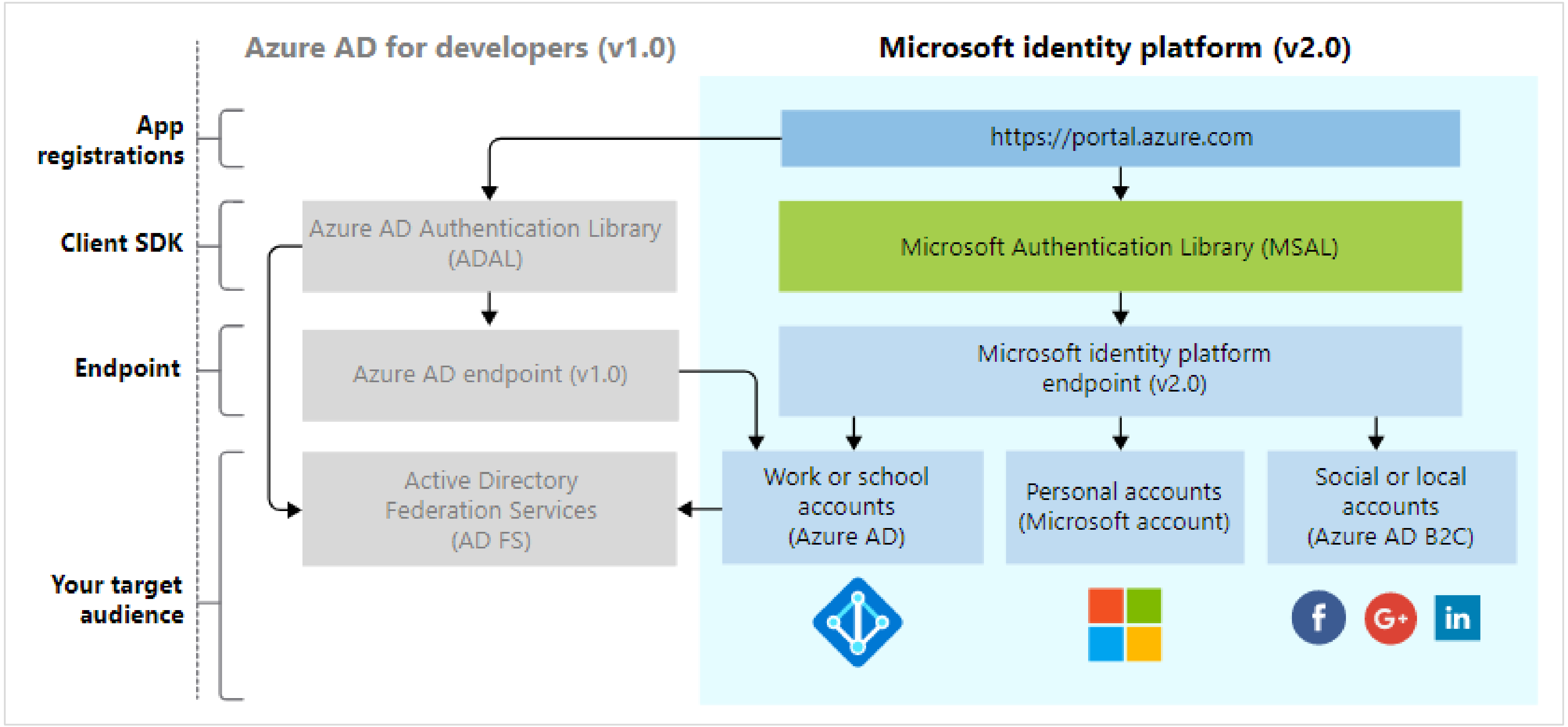
Rob Windsor
rwindsor@paitgroup.com
@robwindsor

About Me

- Lead SharePoint Consultant at PAIT Group
- Microsoft MVP, MCPD, MCT Alumni
- Founder and Past-president of the North Toronto .NET UG

Microsoft Identity Platform

Evolution of the Azure Active Directory (Azure AD) developer platform


Azure AD for developers (v1.0)
Microsoft identity platform (v2.0)
App registrations
Client SDK
Endpoint
Your target audience
https://portal.azure.com
Azure AD Authentication Library (ADAL)
Microsoft Authentication Library (MSAL)
Azure AD endpoint (v1.0)
Microsoft identity platform endpoint (v2.0)
Active Directory Federation Services (AD FS)
Work or school accounts (Azure AD)
Personal accounts (Microsoft account)
Social or local accounts (Azure AD B2C)

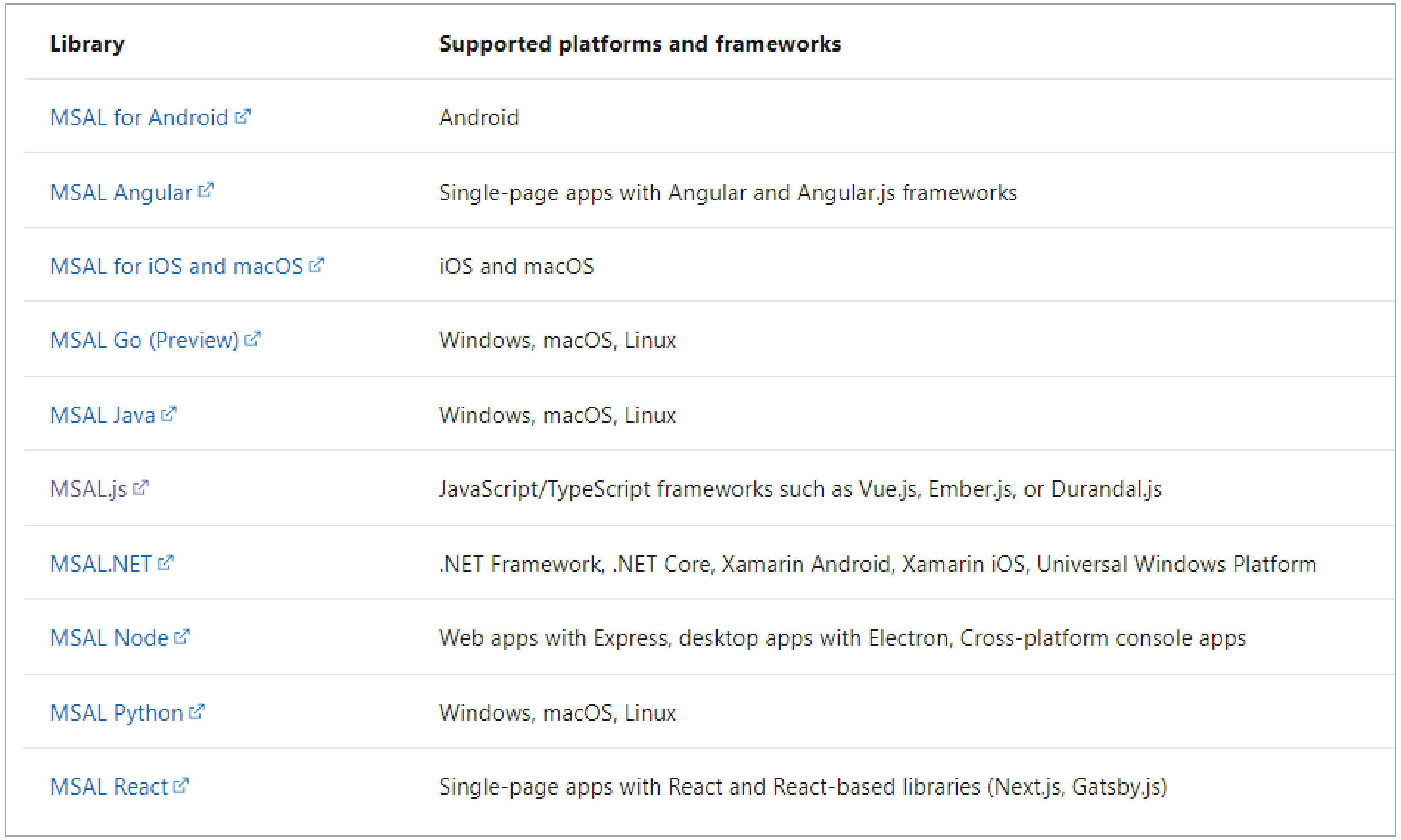
Microsoft Identity Platform Components

- OAuth 2.0 and OpenID Connect standard-compliant auth service
- Open-source libraries
- Application management portal
- Application configuration API and PowerShell
- Developer content

Microsoft Authentication Library (MSAL)

- Enables developers to easily authenticate users and acquire tokens
  - No need to directly use OAuth libraries or protocol
  - Acquires tokens on behalf of a user or application
  - Maintains a token cache and refreshes tokens as needed
  - Helps you to specify audience for app
    - E.g., single tenant or multiple tenant
  - Helps you set up app from configuration files
  - Exposes actionable exceptions, logging, and telemetry

MSAL Languages, Platforms, and Frameworks



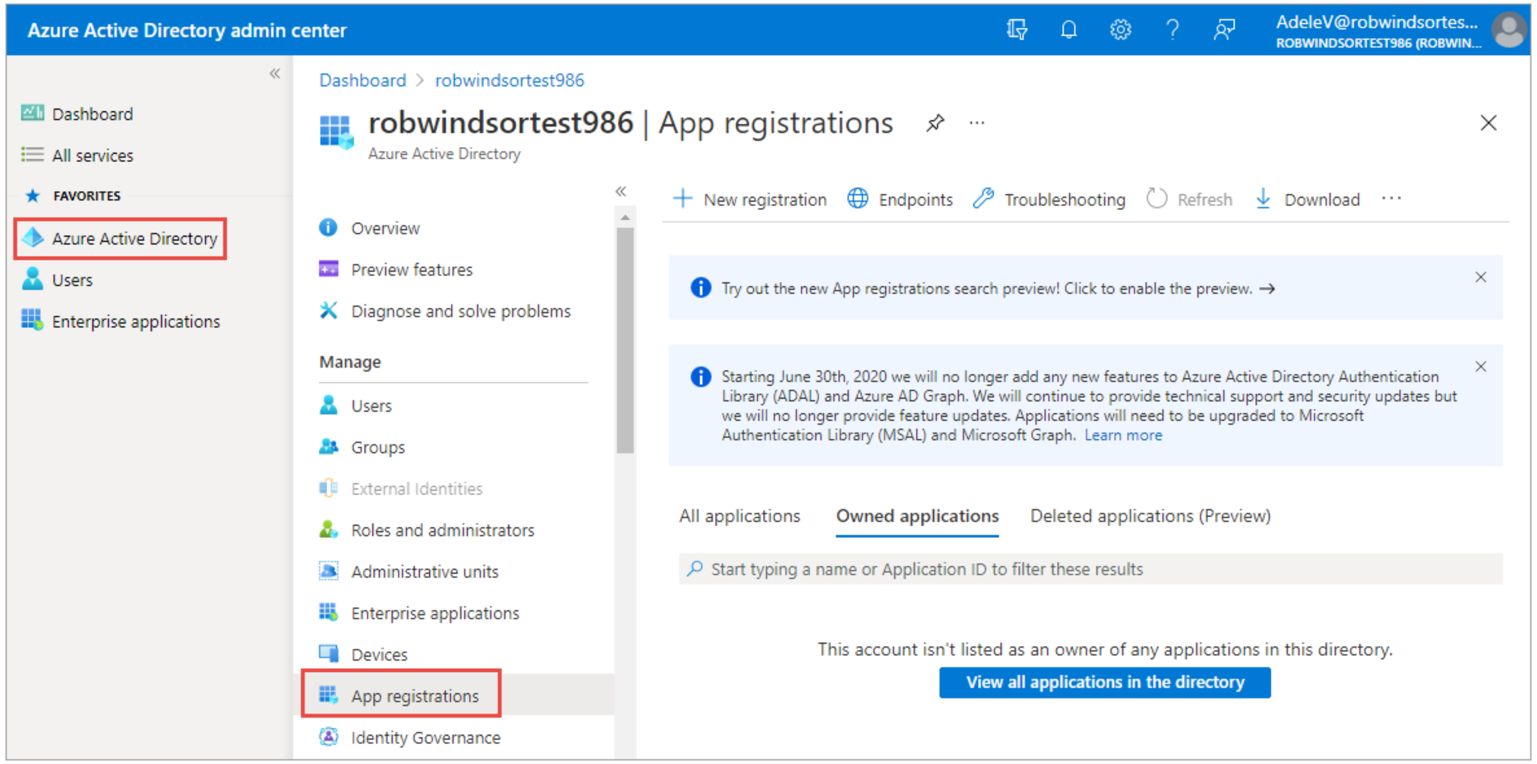
Library	Supported platforms and frameworks
MSAL for Android	Android
MSAL Angular	Single-page apps with Angular and Angular.js frameworks
MSAL for iOS and macOS	iOS and macOS
MSAL Go (Preview)	Windows, macOS, Linux
MSAL Java	Windows, macOS, Linux
MSAL.js	JavaScript/TypeScript frameworks such as Vue.js, Ember.js, or Durandal.js
MSAL.NET	.NET Framework, .NET Core, Xamarin Android, Xamarin iOS, Universal Windows Platform
MSAL Node	Web apps with Express, desktop apps with Electron, Cross-platform console apps
MSAL Python	Windows, macOS, Linux
MSAL React	Single-page apps with React and React-based libraries (Next.js, Gatsby.js)

Microsoft.Identity.Web

- Set of ASP.NET Core libraries that sit on top of MSAL
- Simplifies authentication for web apps and web APIs
- Used in .NET 5.0 web project templates

App Registration

- Azure AD apps registered in Azure portal
- https://portal.azure.com or https://aad.portal.azure.com

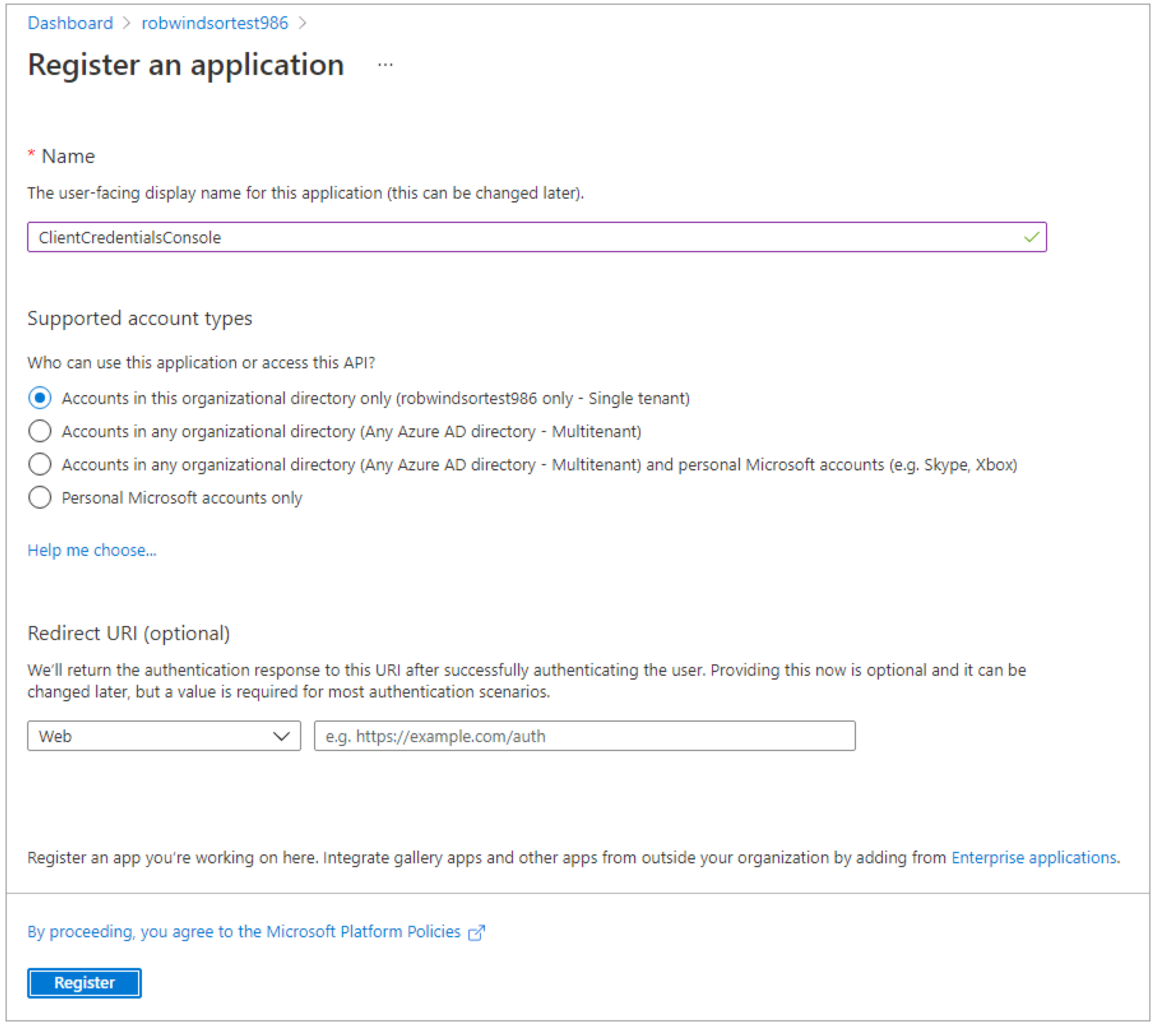


Try out the new App registrations search preview! Click to enable the preview. →

Starting June 30th, 2020 we will no longer add any new features to Azure Active Directory Authentication Library (ADAL) and Azure AD Graph. We will continue to provide technical support and security updates but we will no longer provide feature updates. Applications will need to be upgraded to Microsoft Authentication Library (MSAL) and Microsoft Graph. Learn more

This account isn't listed as an owner of any applications in this directory.

App Registration



Dashboard > robwindsortest986 >

Register an application ···

* Name

The user-facing display name for this application (this can be changed later).

ClientCredentialsConsole ✓

Supported account types

Who can use this application or access this API?

- [✓ Accounts in this organizational directory only (robwindsortest986 only - Single tenant)]
- Accounts in any organizational directory (Any Azure AD directory - Multitenant)
- Accounts in any organizational directory (Any Azure AD directory - Multitenant) and personal Microsoft accounts (e.g. Skype, Xbox)
- Personal Microsoft accounts only

Help me choose...

Redirect URI (optional)

We'll return the authentication response to this URI after successfully authenticating the user. Providing this now is optional and it can be changed later, but a value is required for most authentication scenarios.

Web | e.g. https://example.com/auth

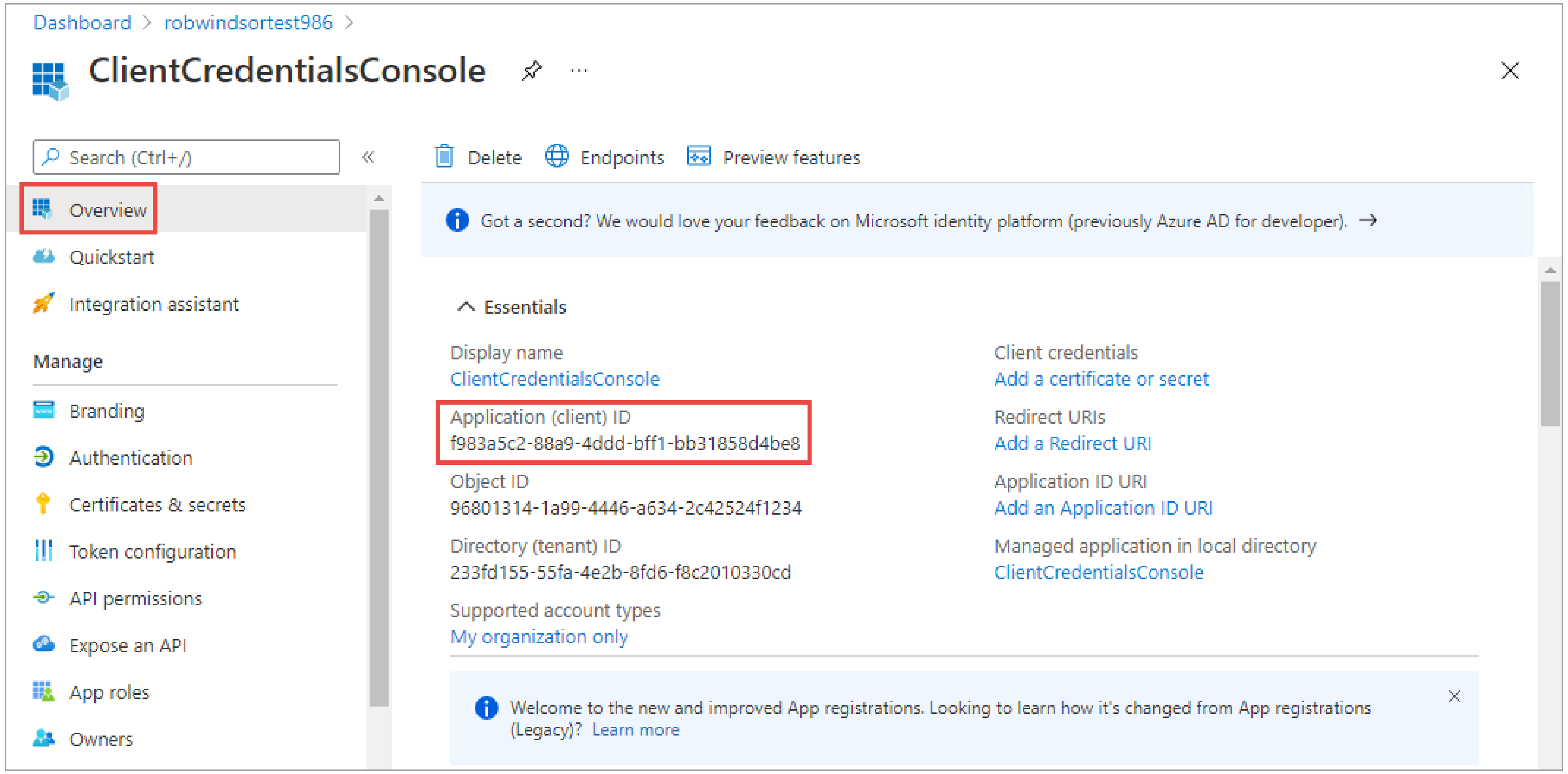
Register an app you're working on here. Integrate gallery apps and other apps from outside your organization by adding from Enterprise applications.

By proceeding, you agree to the Microsoft Platform Policies

Register

App Registration

- All apps uniquely identified by Client ID (a.k.a., Application ID)



Dashboard > robwindsortest986 >

ClientCredentialsConsole

Got a second? We would love your feedback on Microsoft identity platform (previously Azure AD for developer). →

Essentials

Display name	ClientCredentialsConsole
Application (client) ID	f983a5c2-88a9-4ddd-bff1-bb31858d4be8
Object ID	96801314-1a99-4446-a634-2c42524f1234
Directory (tenant) ID	233fd155-55fa-4e2b-8fd6-f8c2010330cd
Supported account types	My organization only
Client credentials	Add a certificate or secret
Redirect URIs	Add a Redirect URI
Application ID URI	Add an Application ID URI
Managed application in local directory	ClientCredentialsConsole

Welcome to the new and improved App registrations. Looking to learn how it's changed from App registrations (Legacy)? Learn more

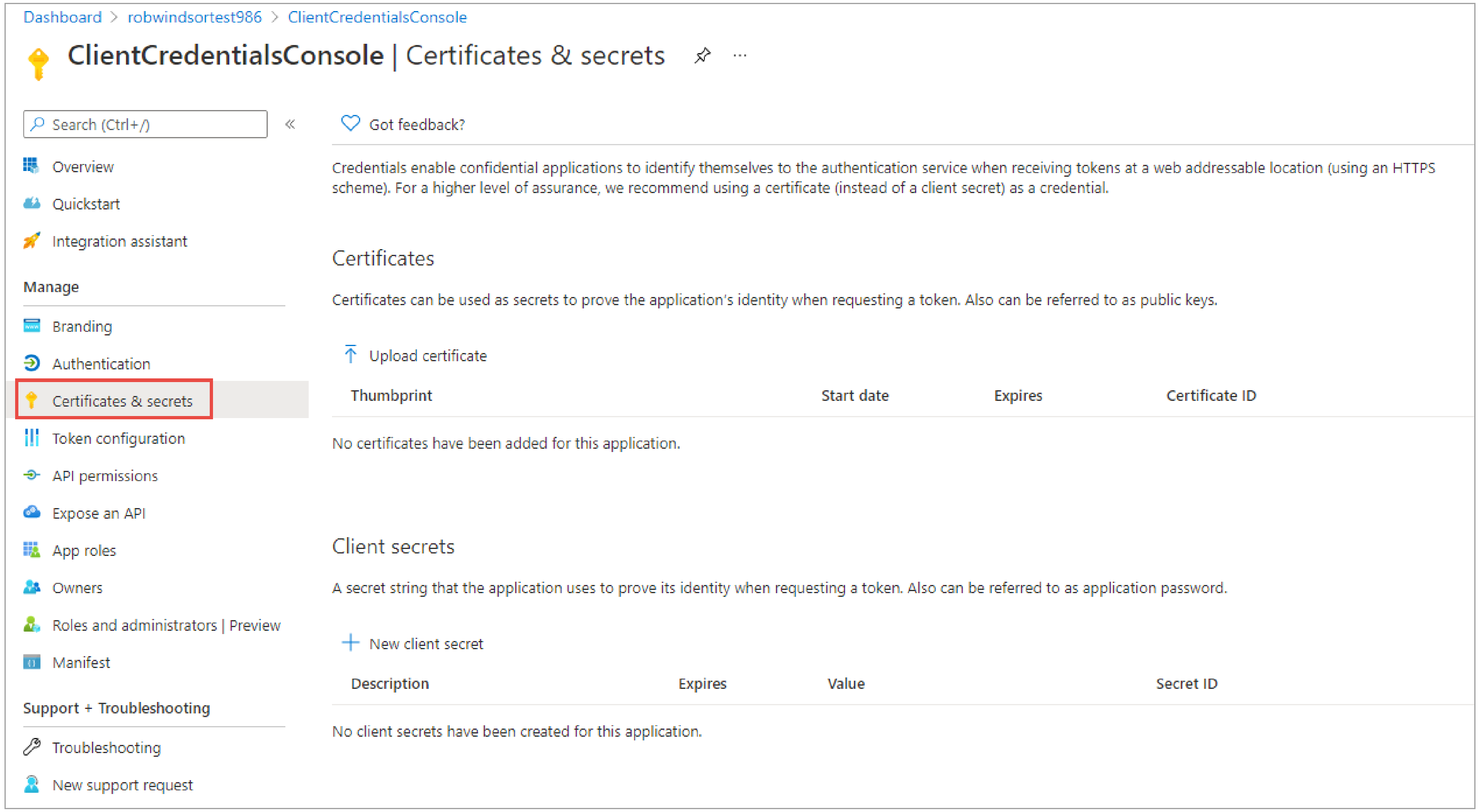
Authentication Flows

- Process used to authenticate with Azure AD depends on
  - Type of application (e.g., web, mobile, desktop)
  - If there is a signed-in user
- These different processes are known as Authentication Flows
- More on authentication flows later

Client Secrets and Certificates

- Some auth flows require Azure AD app to present credentials
    - Analogous to username and password for a user
- Client ID used as username
- Client secrets or certificates used as passwords

Client Secrets and Certificates



Dashboard > robwindsortest986 > ClientCredentialsConsole

ClientCredentialsConsole | Certificates & secrets

Credentials enable confidential applications to identify themselves to the authentication service when receiving tokens at a web addressable location (using an HTTPS scheme). For a higher level of assurance, we recommend using a certificate (instead of a client secret) as a credential.

Certificates

Certificates can be used as secrets to prove the application's identity when requesting a token. Also can be referred to as public keys.

Thumbprint	Start date	Expires	Certificate ID

No certificates have been added for this application.

Client secrets

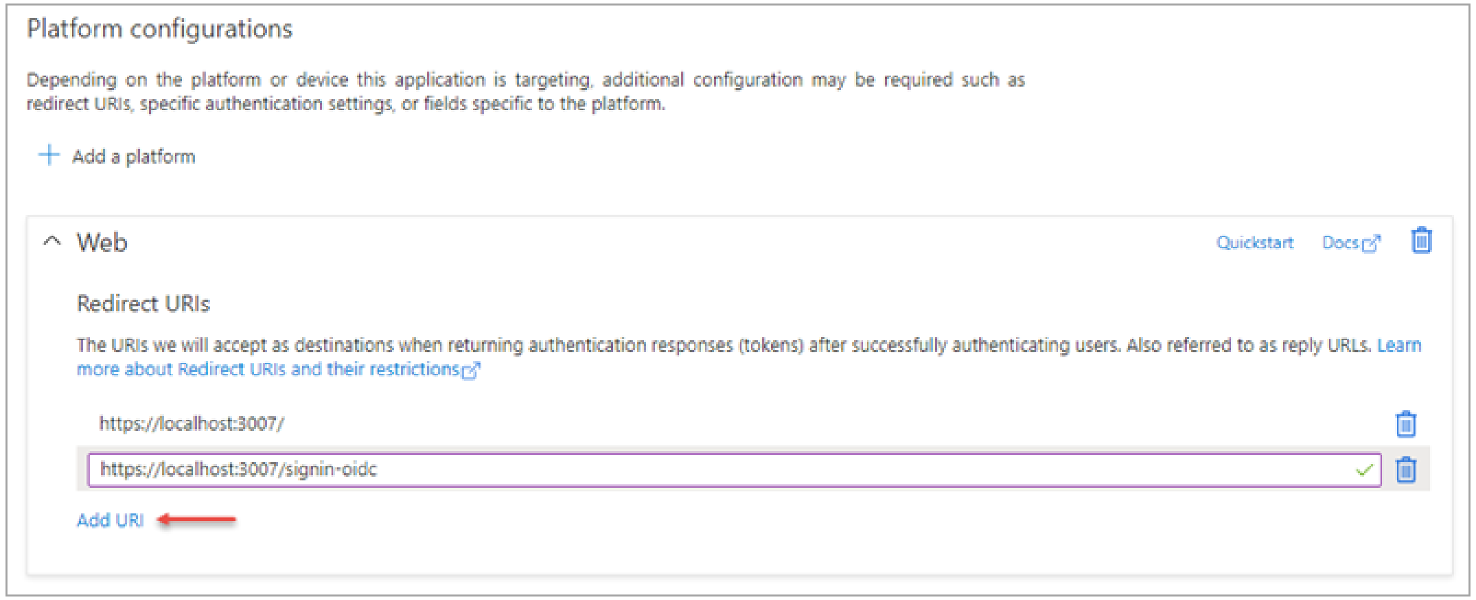
A secret string that the application uses to prove its identity when requesting a token. Also can be referred to as application password.

Description	Expires	Value	Secret ID

No client secrets have been created for this application.

Redirect URIs (Reply URLs)

- Azure AD can only send tokens to URIs registered with Azure AD app
- Multiple URIs may be registered with app
- Wildcards may not be used in URI



Platform configurations

Depending on the platform or device this application is targeting, additional configuration may be required such as redirect URIs, specific authentication settings, or fields specific to the platform.

Redirect URIs

The URIs we will accept as destinations when returning authentication responses (tokens) after successfully authenticating users. Also referred to as reply URLs. Learn more about Redirect URIs and their restrictions

https://localhost:3007/

https://localhost:3007/signin-oidc

Permission Requests

- Apps request permission to use functionality in APIs
- Static
  - Permission requests (scopes) defined in Azure portal
- Dynamic
  - Permission requests (scopes) defined in code
  - Enables incremental permission requests
    - Additional permissions can be requested as needed

Permission Types

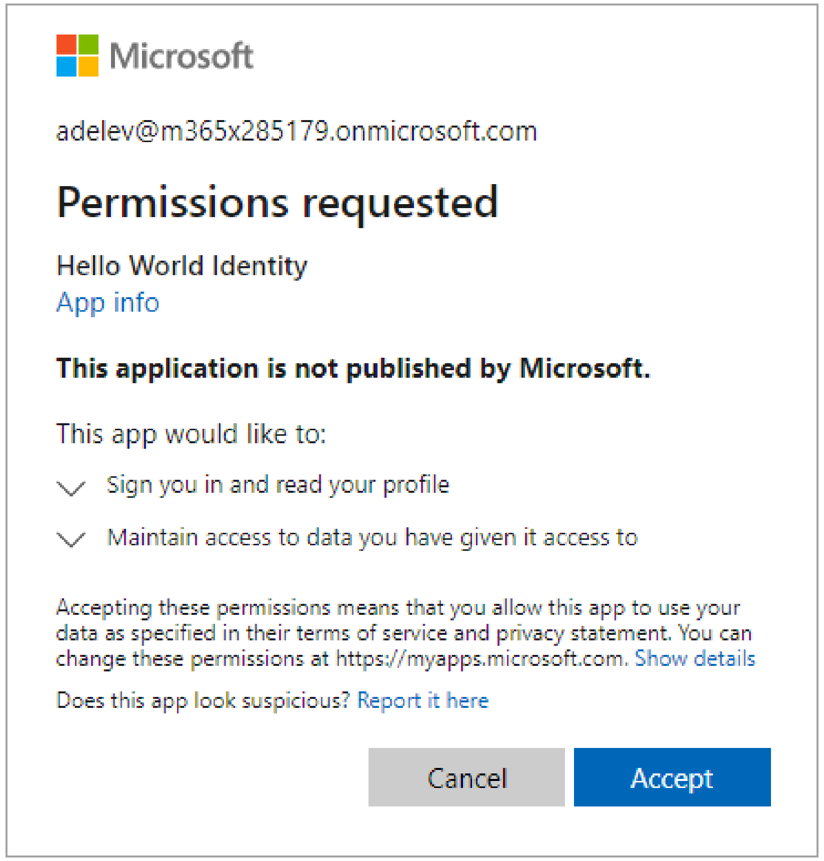
- Delegated permissions
  - Used by apps that have a signed-in user
  - App is delegated with permission to act as sign-in user
  - Effective permissions are intersection of user and app permissions
  - Permission requests can be granted by user or by admin
- Application permissions
  - Generally used by apps that do not have a signed-in user
  - Can also be used by apps that have a signed-in user who may not have permission necessary to interact with resource
  - Effective permissions are the same as the app permissions
  - User permissions disregarded
  - Permission requests can only be granted by admin

Permission Scopes

- Define the set of permissions requested by an app
- Represented by two-part string
  - Resource identifier
  - Permission requested
- Examples
  - https://graph.microsoft.com/Calendars.Read
  - https://contoso.sharepoint.com/Sites.Read.All
- Azure AD assumes Microsoft Graph if resource identifier not specified
  - Calendars.Read is the same as https://graph.microsoft.com/Calendars.Read

Permission Consent

User consent



Microsoft

adelev@m365x285179.onmicrosoft.com

Permissions requested

Hello World Identity
App info

This application is not published by Microsoft.

This app would like to:

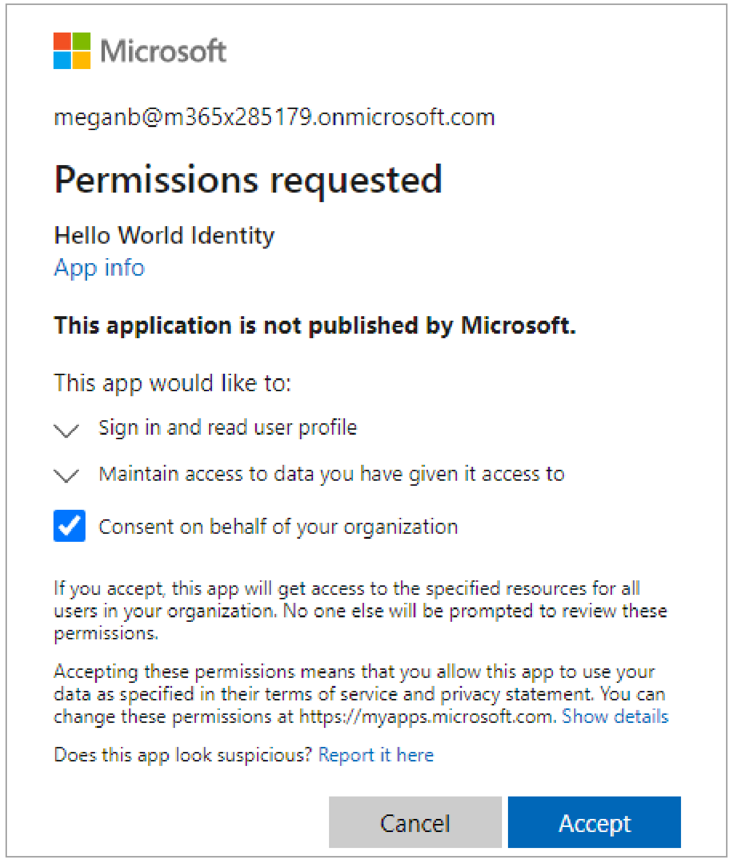
Sign you in and read your profile

Maintain access to data you have given it access to

Accepting these permissions means that you allow this app to use your data as specified in their terms of service and privacy statement. You can change these permissions at https://myapps.microsoft.com. Show details

Does this app look suspicious? Report it here

Admin consent



Microsoft

meganb@m365x285179.onmicrosoft.com

Permissions requested

Hello World Identity
App info

This application is not published by Microsoft.

This app would like to:

Sign in and read user profile

Maintain access to data you have given it access to

✓ Consent on behalf of your organization

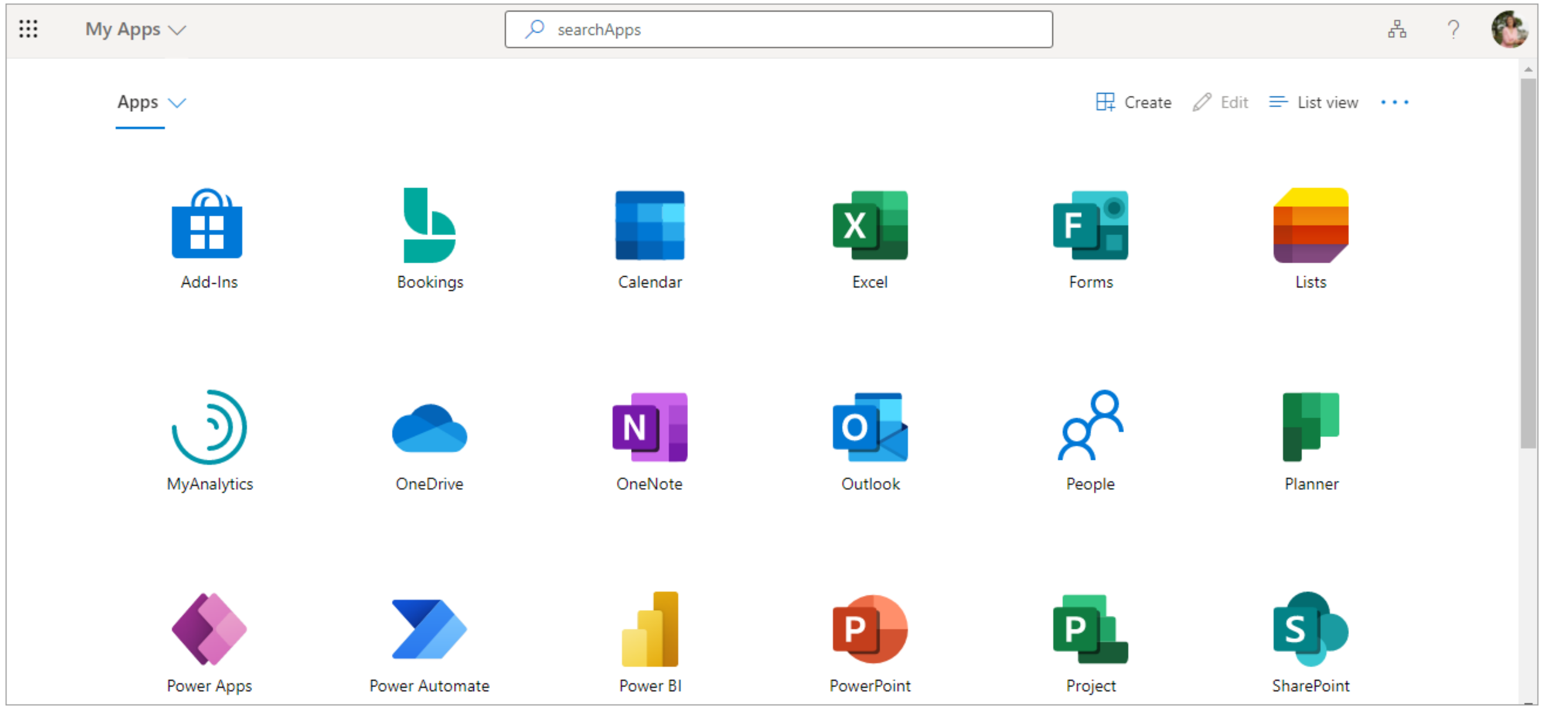
If you accept, this app will get access to the specified resources for all users in your organization. No one else will be prompted to review these permissions.

Accepting these permissions means that you allow this app to use your data as specified in their terms of service and privacy statement. You can change these permissions at https://myapps.microsoft.com. Show details

Does this app look suspicious? Report it here

My Applications

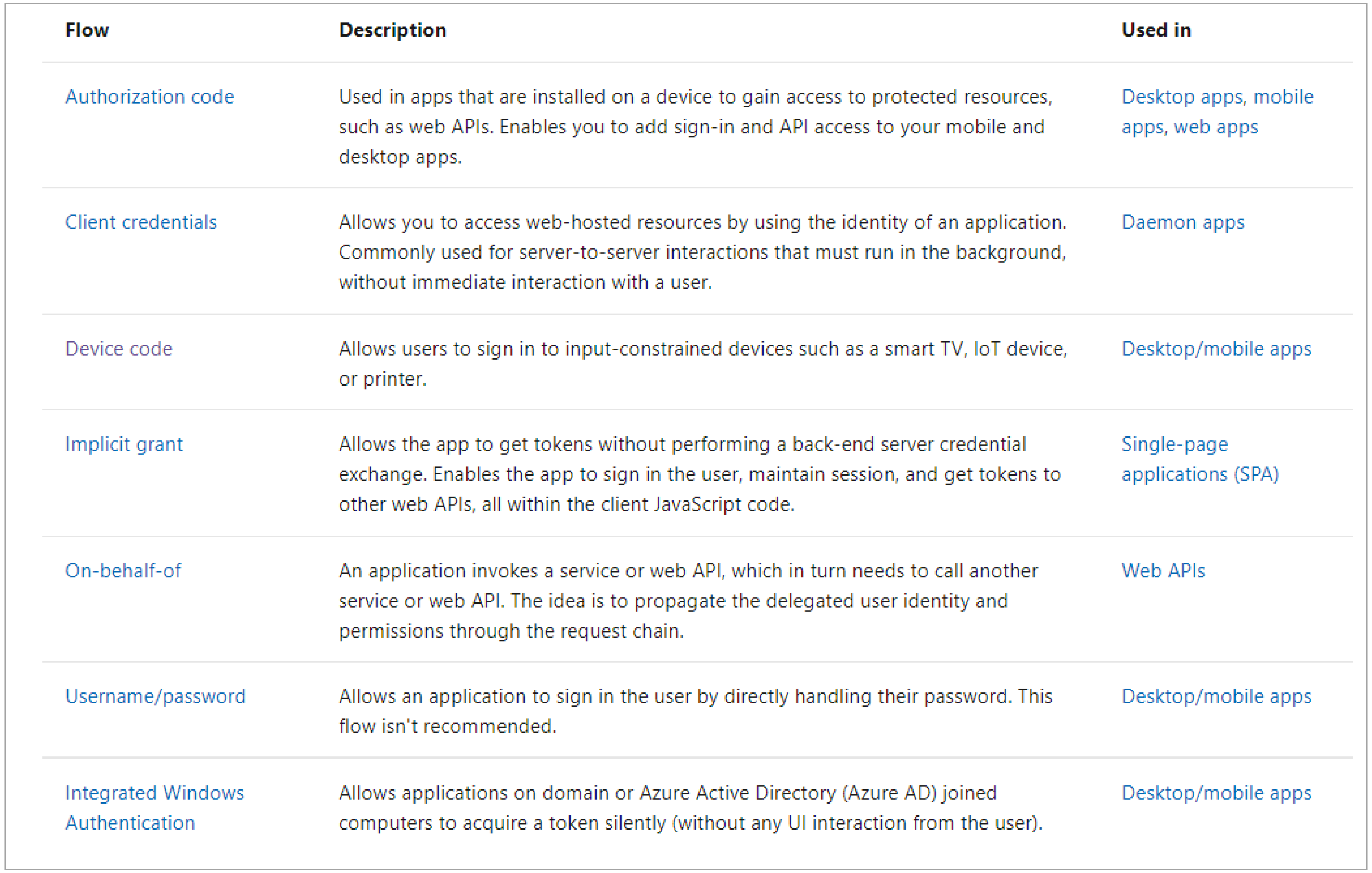
- Users can manage their applications using the My Apps portal
- https://myapplications.microsoft.com/



Tokens

- Implemented as JSON Web Tokens (“JOTs”)
- Access token
    - Short-lived
    - Enable clients to securely call APIs protected by Azure AD
- Refresh token
    - Long-lived
    - Used to silently request new access tokens
- ID token
    - Used to identify user

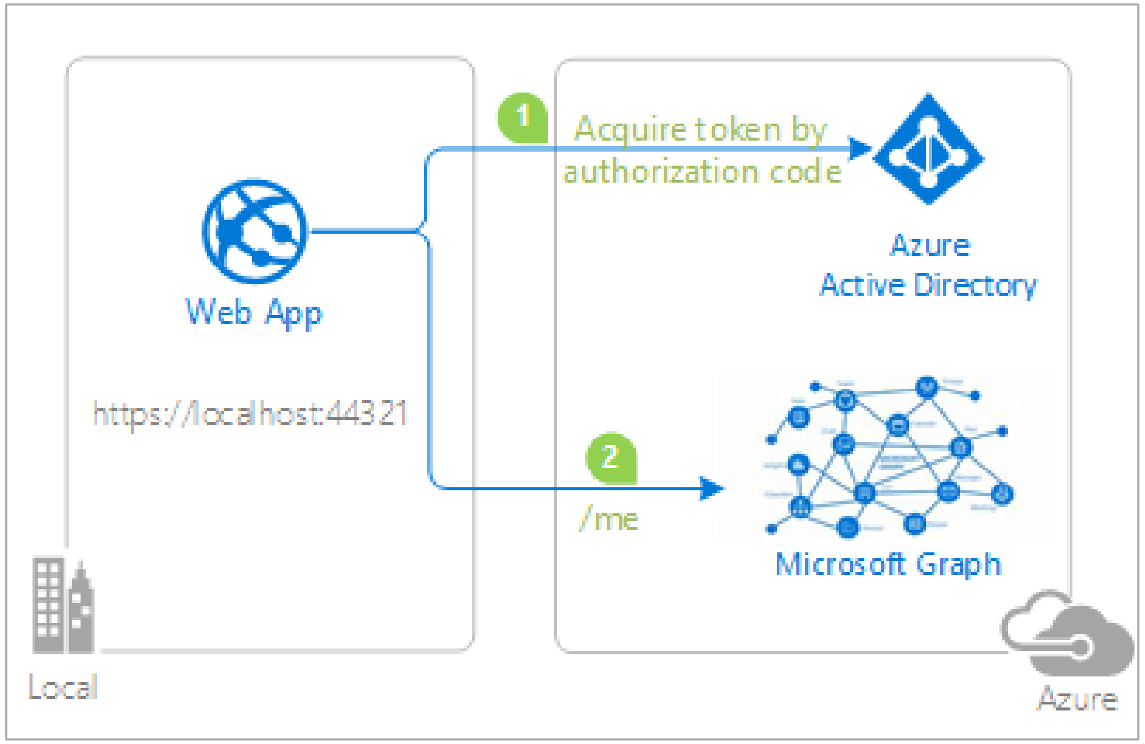
Authentication Flows



Flow	Description	Used in
Authorization code	Used in apps that are installed on a device to gain access to protected resources, such as web APIs. Enables you to add sign-in and API access to your mobile and desktop apps.	Desktop apps, mobile apps, web apps
Client credentials	Allows you to access web-hosted resources by using the identity of an application. Commonly used for server-to-server interactions that must run in the background, without immediate interaction with a user.	Daemon apps
Device code	Allows users to sign in to input-constrained devices such as a smart TV, IoT device, or printer.	Desktop/mobile apps
Implicit grant	Allows the app to get tokens without performing a back-end server credential exchange. Enables the app to sign in the user, maintain session, and get tokens to other web APIs, all within the client JavaScript code.	Single-page applications (SPA)
On-behalf-of	An application invokes a service or web API, which in turn needs to call another service or web API. The idea is to propagate the delegated user identity and permissions through the request chain.	Web APIs
Username/password	Allows an application to sign in the user by directly handling their password. This flow isn't recommended.	Desktop/mobile apps
Integrated Windows Authentication	Allows applications on domain or Azure Active Directory (Azure AD) joined computers to acquire a token silently (without any UI interaction from the user).	Desktop/mobile apps

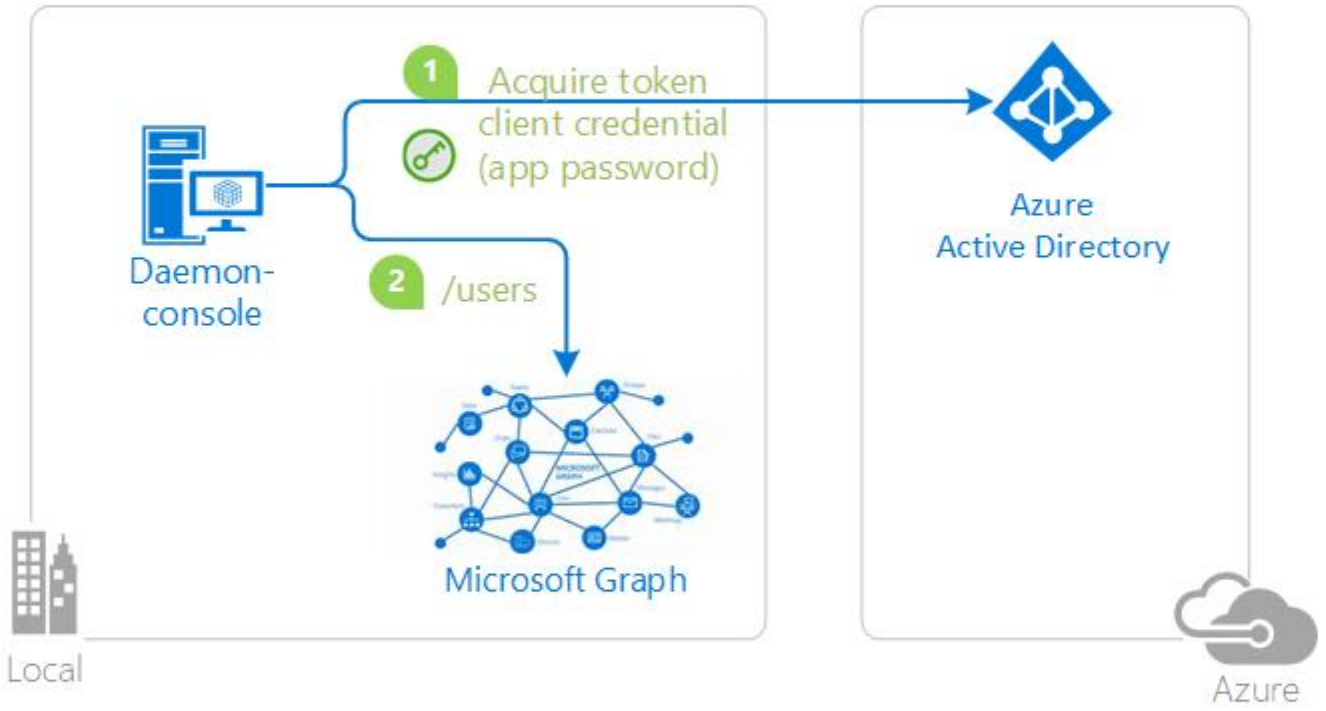
Flows – Authorization Code

- Users receive authorization code on sign-in
- Authorization code is exchanged for access token



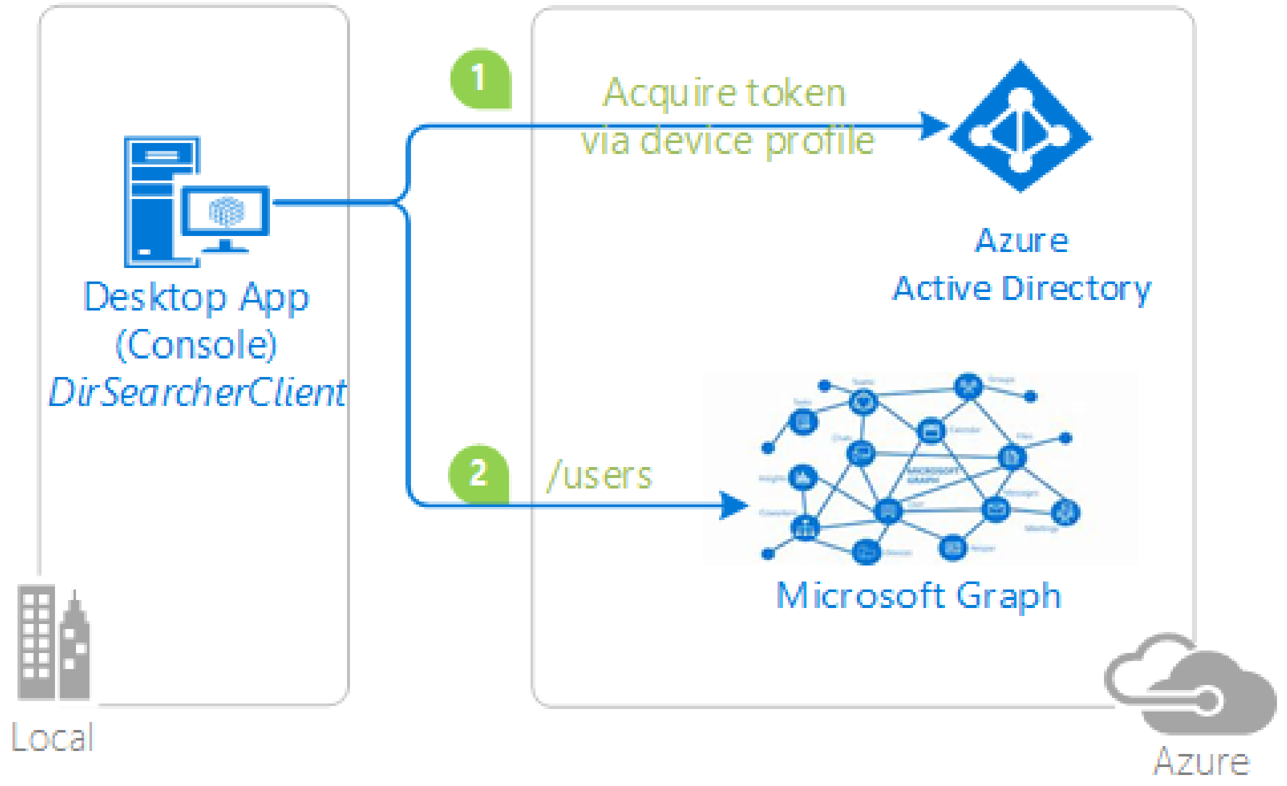
Flows – Client Credentials

- Authentication done using Client ID and Client Secret or Client ID and certificate



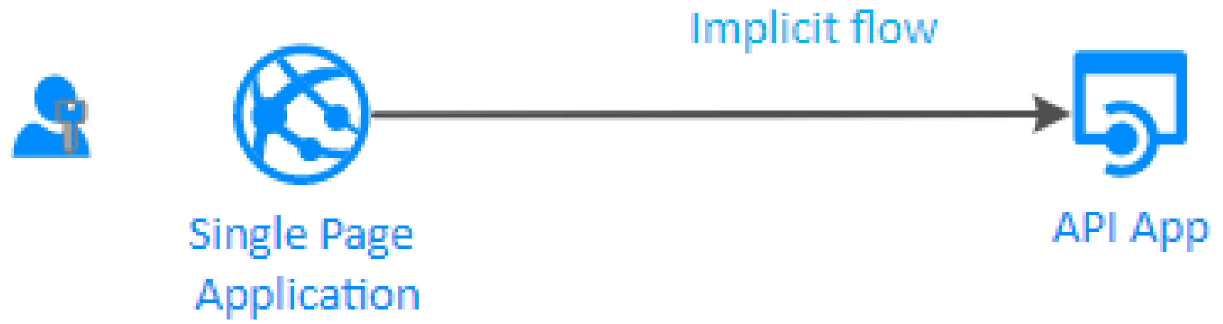
Flows – Device Code

- App provides user with code and directs user to URL
- User opens browser to URL, enters code, and authenticates normally



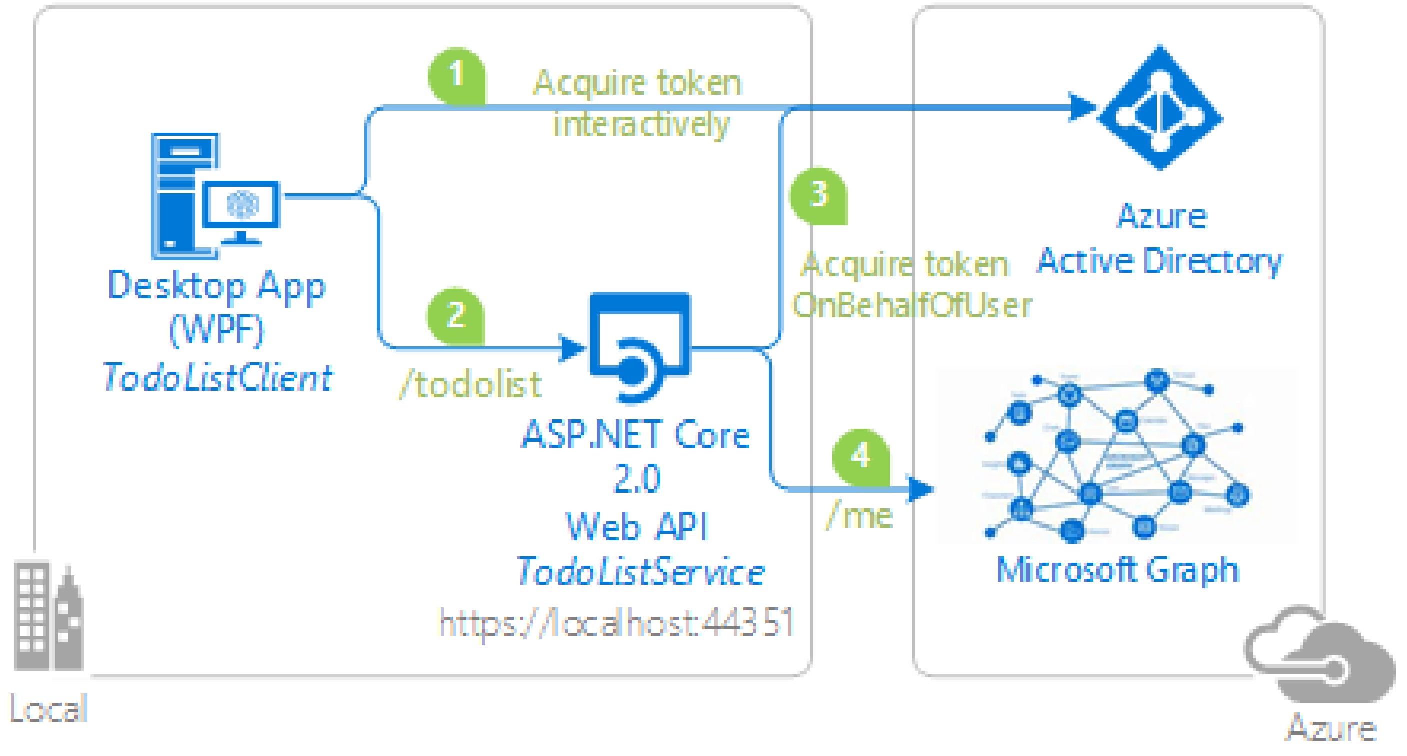
Flows – Implicit Grant

- Legacy
- New SPAs should use authorization code flow



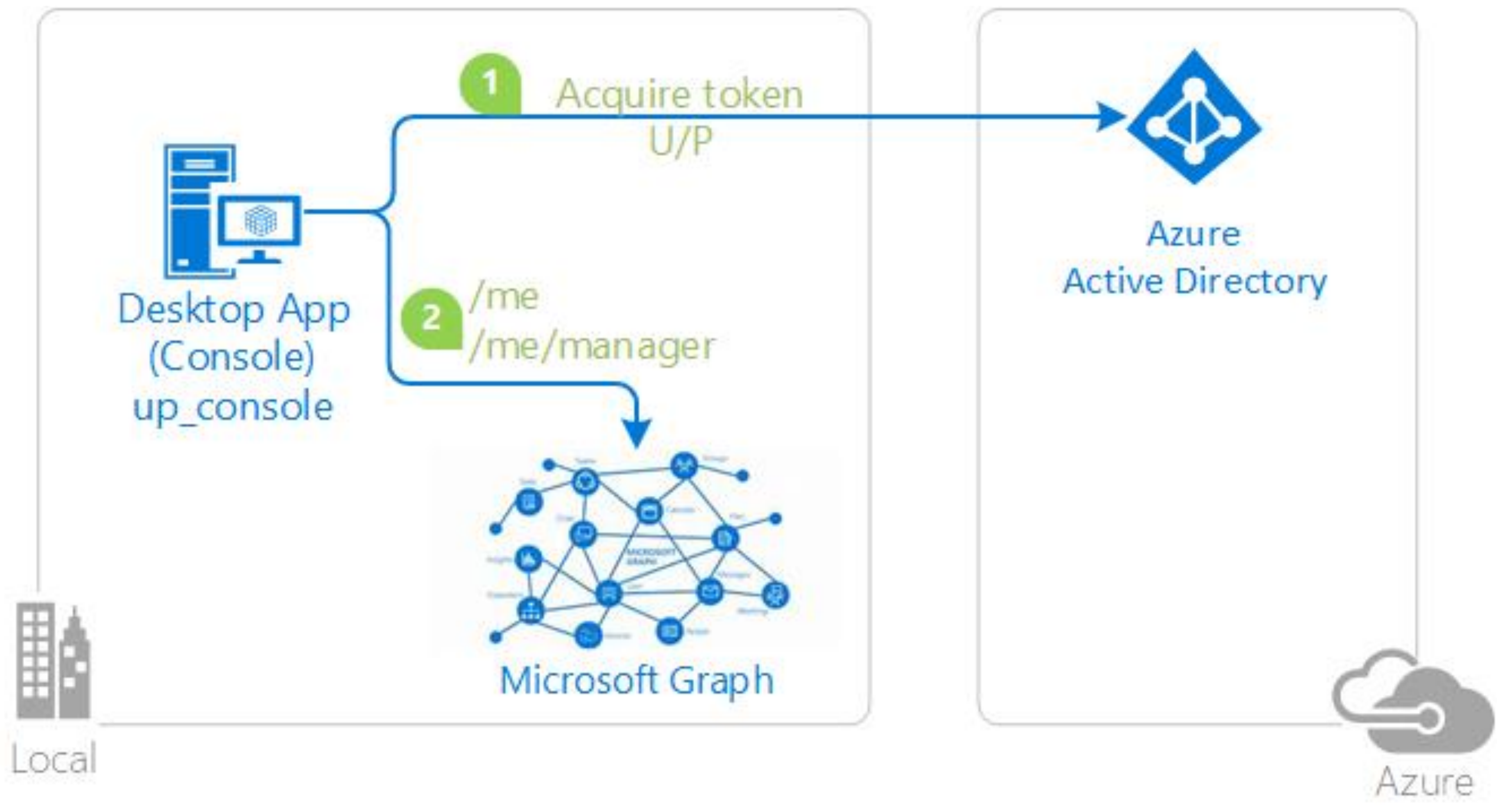
Flows – On-behalf-of

- Access token for one resource used to get access token for second resource



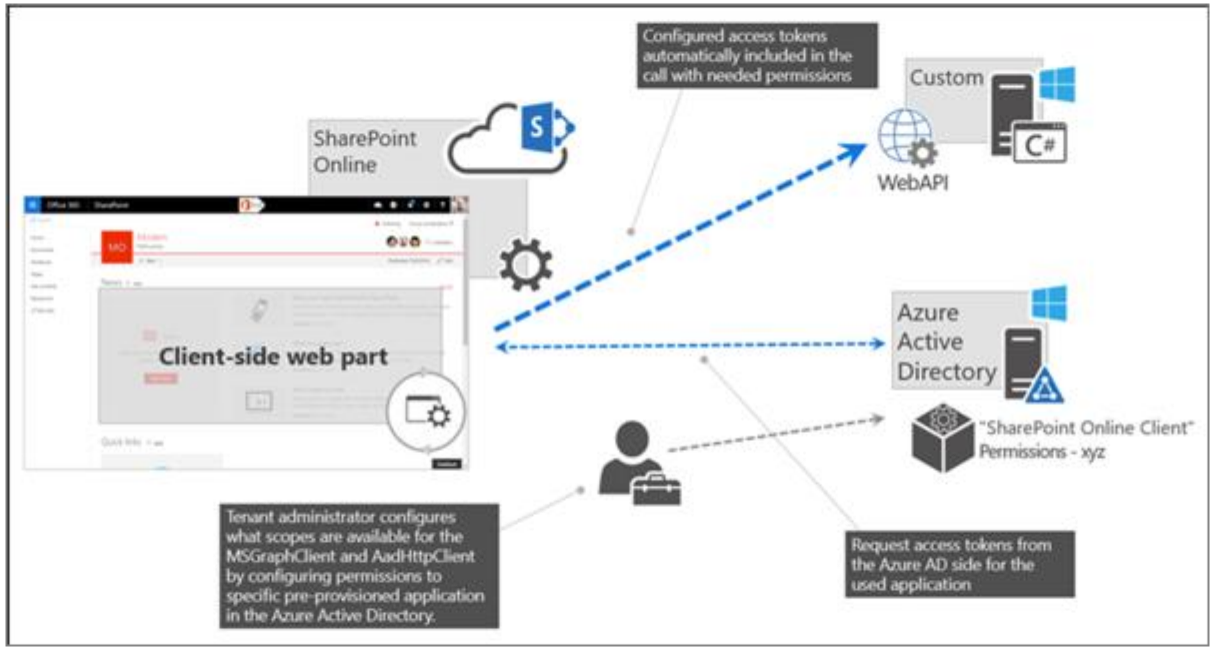
Flows – Username and Password

- Authentication done using username and password



SharePoint Framework

- AadHttpClient
  - @microsoft/sp-http package
- Abstract the token acquisition from SPFx support for Azure AD


Configured access tokens automatically included in the call with needed permissions
Custom
C#
WebAPI
SharePoint Online
Client-side web part
Azure Active Directory
"SharePoint Online Client" Permissions - xyz
Tenant administrator configures what scopes are available for the MSGraphClient and AadHttpClient by configuring permissions to specific pre-provisioned application in the Azure Active Directory.
Request access tokens from the Azure AD side for the used application

SharePoint Framework – Microsoft Graph

- MSGraphClient
  - @microsoft/sp-http package
- Extends AadHttpClient for use with Microsoft Graph
- Abstract the token acquisition from SPFx support for Azure AD
- Wraps the Microsoft Graph JavaScript SDK and initializes with one line

- Can optionally use with Microsoft Graph Type Declarations
  - @microsoft/microsoft-graph-types package
  - Work with strongly-typed objects representing Graph data

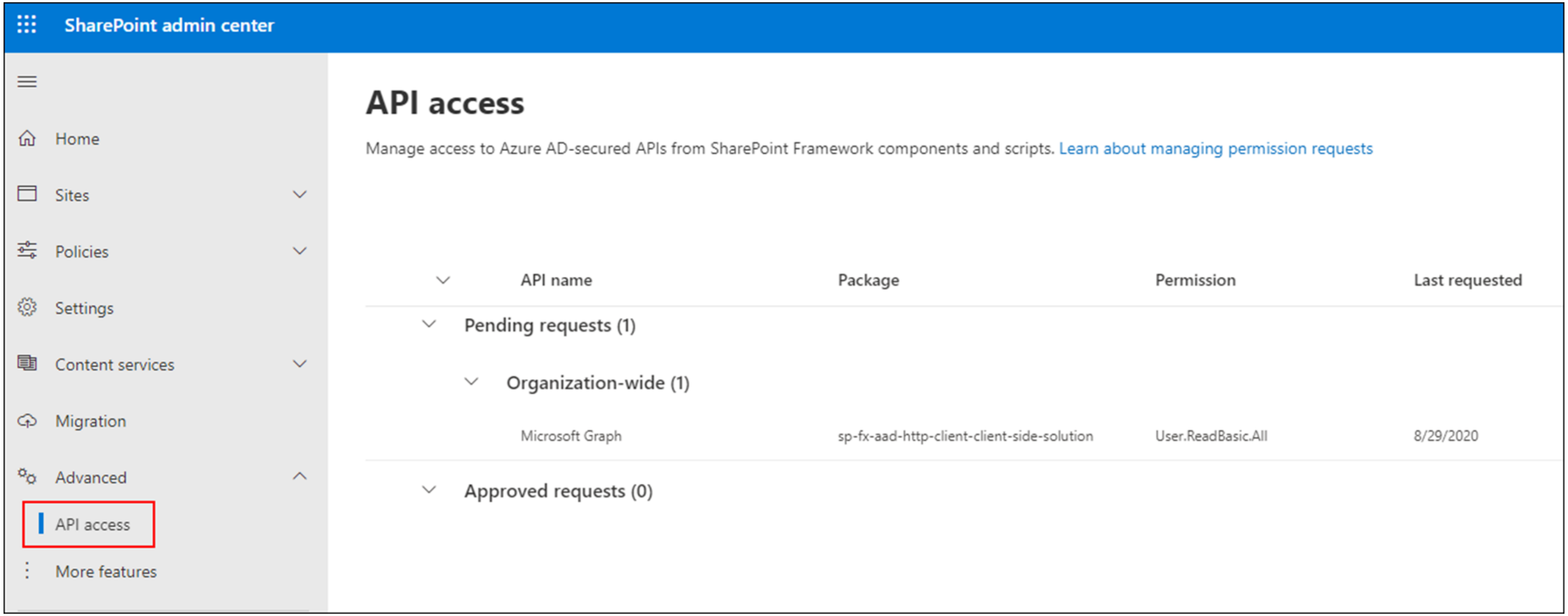
SPFx Solutions Declare Permission Requests

- Do not need to register Azure AD app when using AadHttpClient or MSGraphClient
  - Microsoft does this for you
- Your project declares the permissions it requires
  - webApiPermissionRequests setting in package-solution.json
- Tenant administrator grants or rejects permission requests using API Management page in SharePoint Online administration portal
- Granting permission request updates Azure AD app created by Microsoft

SPFx Solutions Declare Permission Requests

```
{
  "$schema": "https://developer.microsoft.com/json-schemas/spfx-build/package-solution.schema.json",
  "solution": {
    "name": "MSGraphClient Web Part",
    // ....
    "isDomainIsolated": false,
    "webApiPermissionRequests": [
      {
        "resource": "Microsoft Graph",
        "scope": "Group.Read.All"
      }
    ]
  },
  "paths": {
    "zippedPackage": "solution/ms-graph-client.sppkg"
  }
}
```

Approve / Reject with SharePoint Online API Management Page

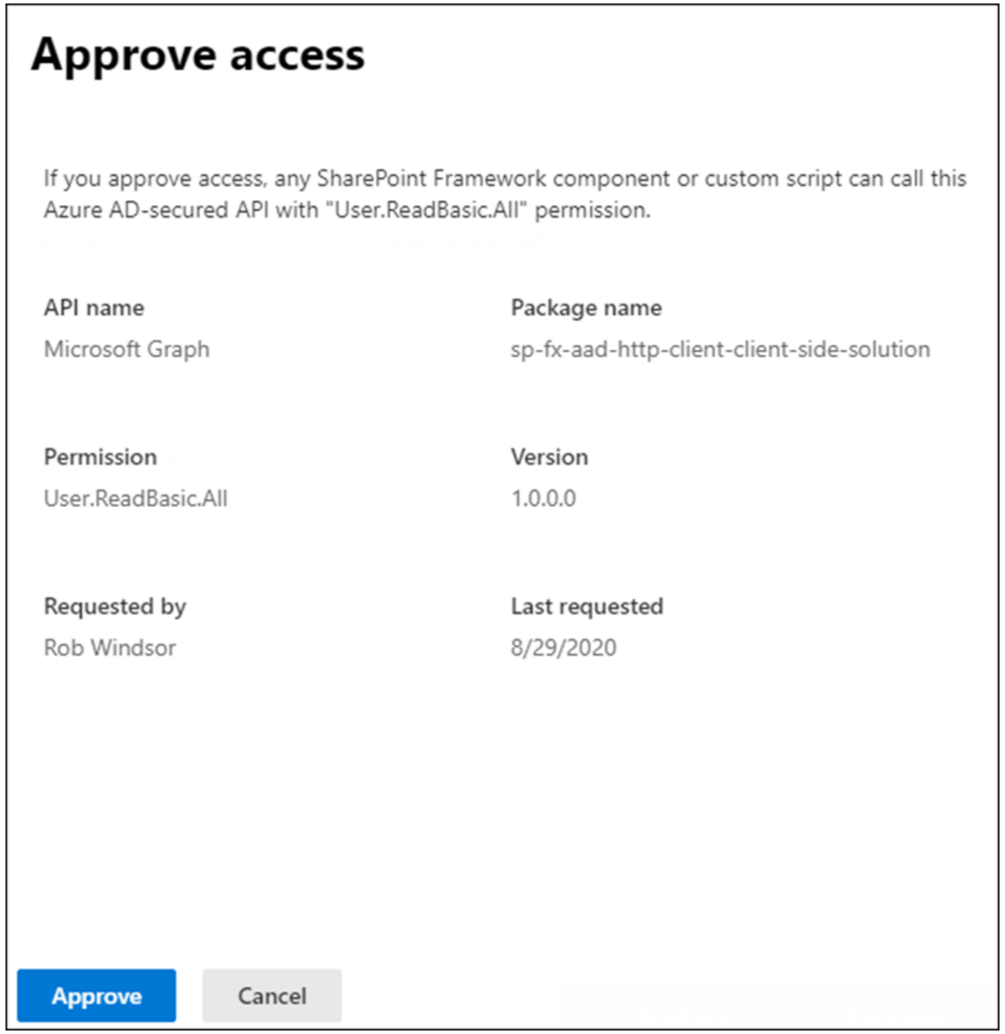


API access

Manage access to Azure AD-secured APIs from SharePoint Framework components and scripts. Learn about managing permission requests

API name	Package	Permission	Last requested
Pending requests (1)			
Organization-wide (1)			
Microsoft Graph	sp-fx-aad-http-client-client-side-solution	User.ReadBasic.All	8/29/2020
Approved requests (0)			

Approve / Reject with SharePoint Online API Management Page



Approve access

If you approve access, any SharePoint Framework component or custom script can call this Azure AD-secured API with "User.ReadBasic.All" permission.

API name	**Package name**
Microsoft Graph	sp-fx-aad-http-client-client-side-solution
Permission	**Version**
User.ReadBasic.All	1.0.0.0
Requested by	**Last requested**
Rob Windsor	8/29/2020

Resources

- Documentation
  - https://docs.microsoft.com/en-us/azure/active-directory/develop/
- Microsoft Learn: Implement Microsoft identity
  - https://docs.microsoft.com/en-us/learn/paths/m365-identity-associate/

Thank You

- Big thanks to the organizers, sponsors, and you for making this event possible
- Please fill out your evaluation
- Please keep in touch

✉ rwindsor@paitgroup.com

@robwindsor

https://blogs.msmvps.com/windsor