Azure
COMMUNITY
CONFERENCE

Azure Community Conference 2021

India's largest Azure Conference

#AzConfDev

Azure COMMUNITY CONFERENCE

Nandeesh Swami

Program Manager II, Identity & Security, Microsoft

#AzConfDev

Azure COMMUNITY CONFERENCE

The Power of Azure Graph

Nandeesh Swami

#AzConfDev

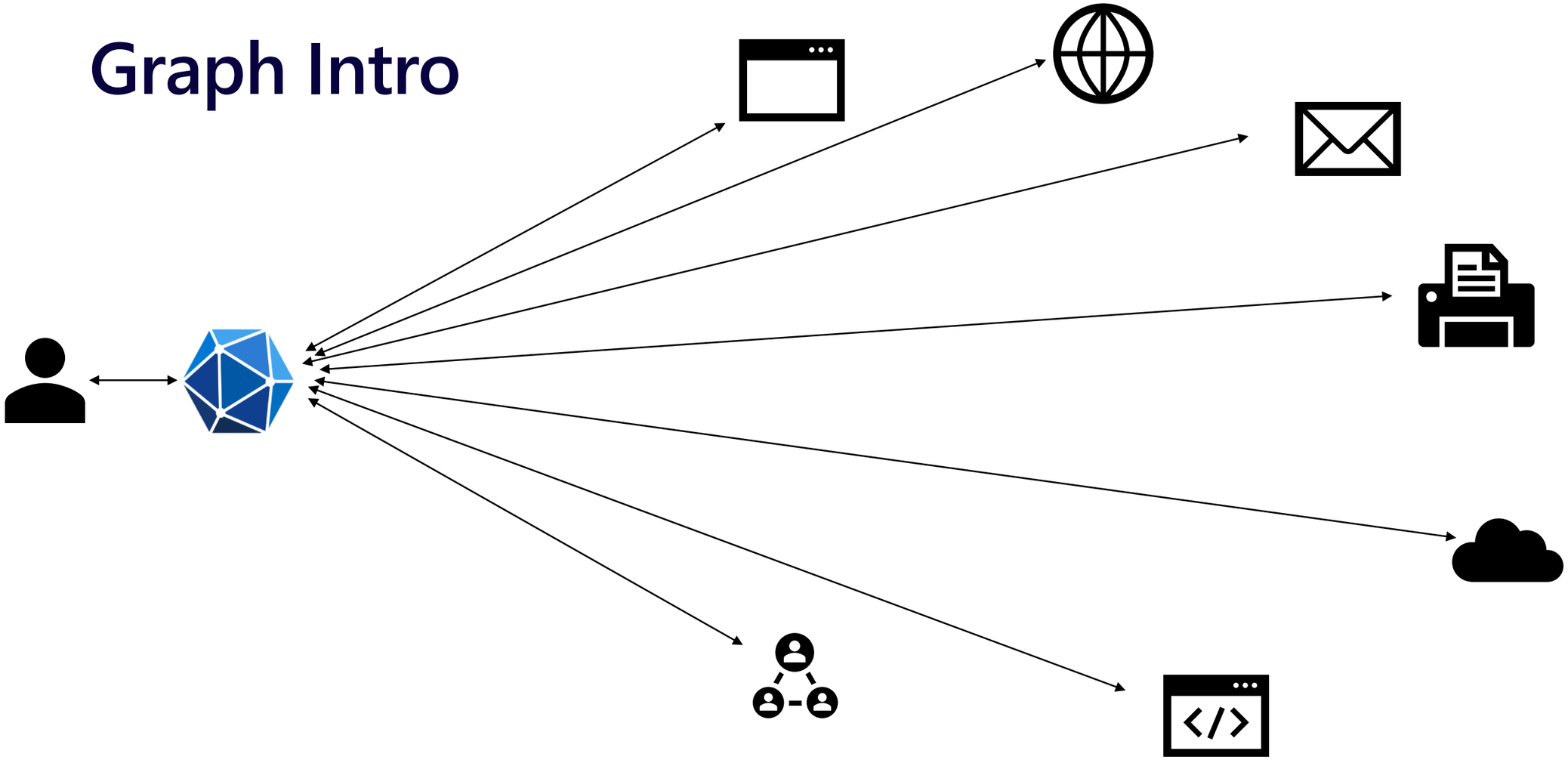
Agenda

1. Graph- Intro
2. Permissions & Consent
3. Graph Explorer
4. Graph Toolkit
5. Coding with Advanced Graph Queries
6. Pagination & more
7. How to be a Graph Pro Developer.

Ok, so what is Graph?

Gateway to all of Microsoft 365 Applications.

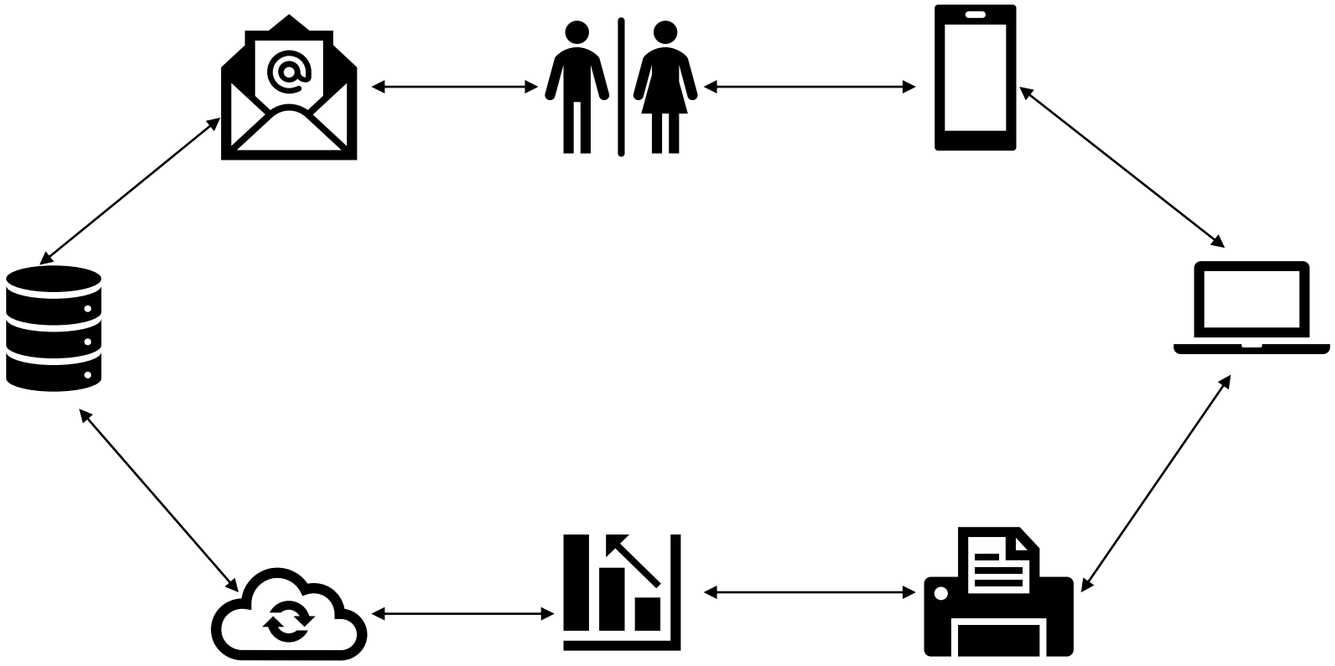
Graph Intro



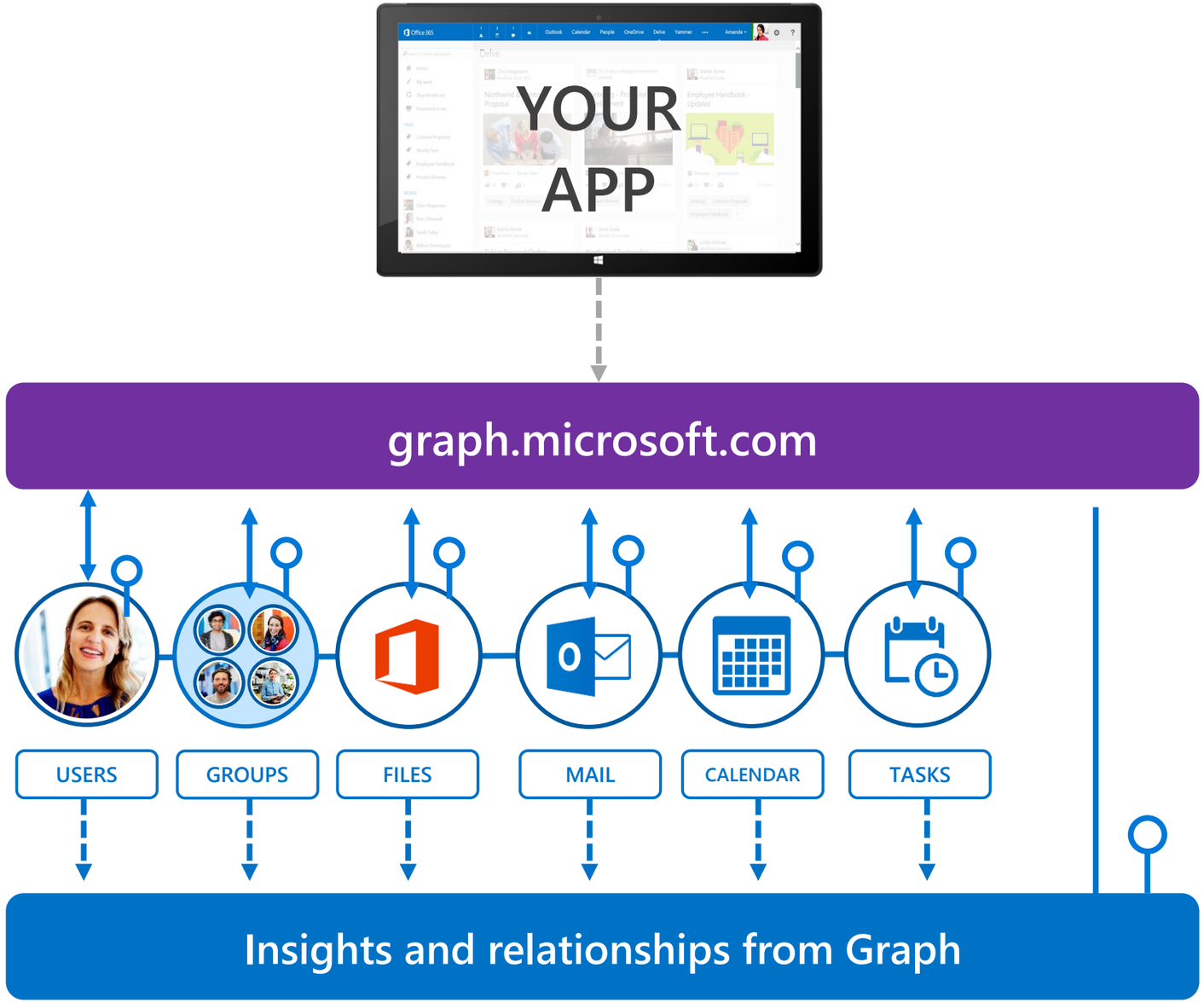
Language agonistic Gateway to data & cloud

Graph Intro

All the resources are connected, and Graph API is a unified endpoint for all of Productivity, Identity and Security Datasets.



Popular Data Sets



Data Sets

- Activities
- Attachments
- Audits
- Calendar
- Categories
- Charts
- Classes
- Contacts
- Conversations
- Cross-device experiences
- Customer booking
- Device configuration
- Device management
- Domains
- Education
- Events
- Files
- Financials
- Groups
- Identity
- Lists
- Mail
- Messages
- Notes
- Notifications
- Pages
- Places
- Plans
- Reports
- Schools
- Search
- Secure score
- Security alerts
- Sites
- Social
- Subscriptions
- Tasks
- Teams
- Threat intelligence
- Users
- Workbooks

• • • and many, many more

Permissions & Consent

- User proving their identity to Microsoft Identity(Authentication)
- Applications need Authorization to be allowed to call APIs

Delegated Permission: On Behalf of the user
Application Permission: Daemon type of Apps

Applications specify the range of operation they REQUIRE by requesting a scope

Application must follow LEAST Privileged Approach

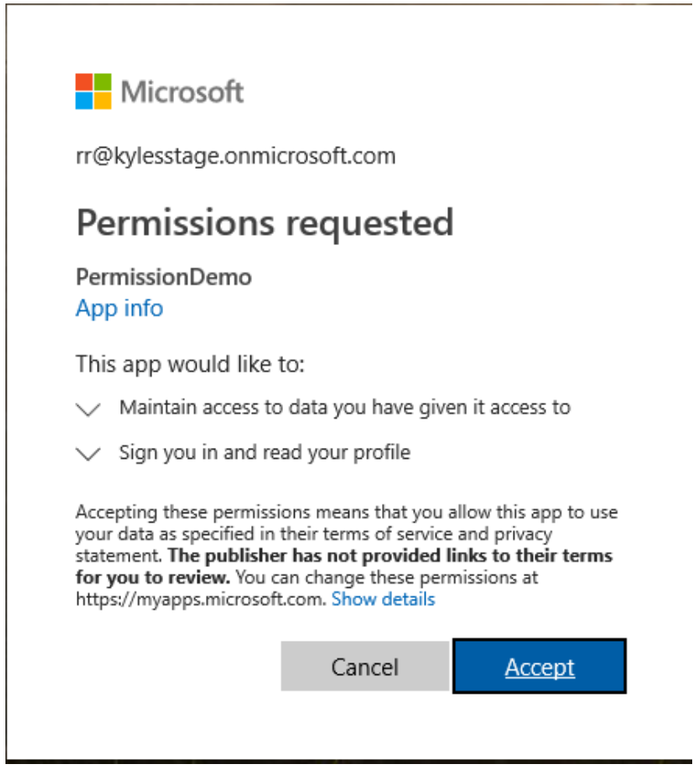
Admin Consent and User Consent

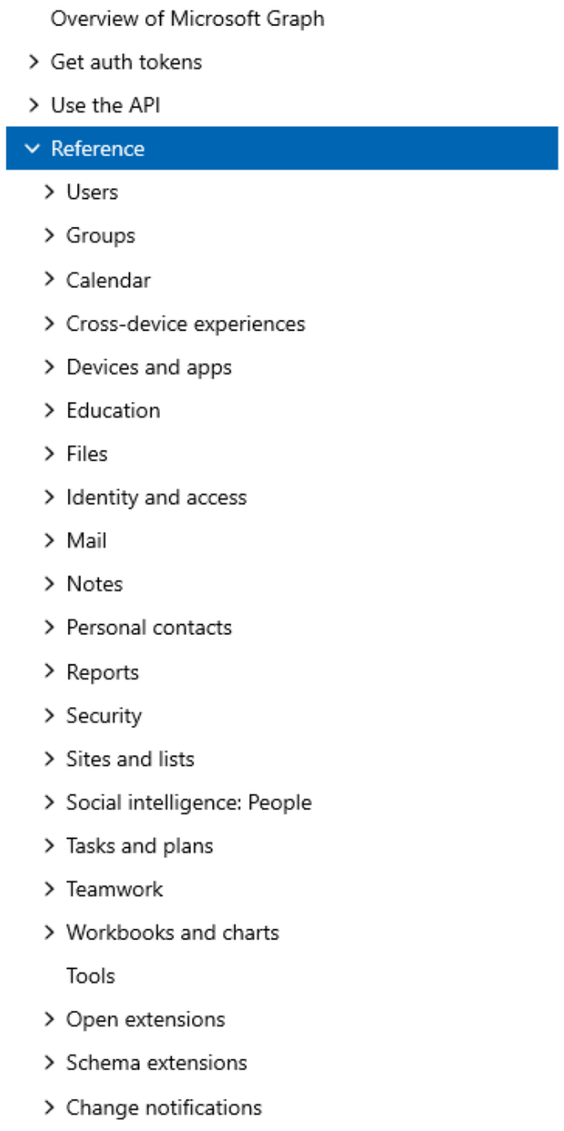
Effective Permissions->The intersection of app permissions and user capabilities

Requesting Scope(Permission)

```
myMSALObj.acquireTokenSilent("User.Read")

if (requiresInteraction(error.errorCode)) {
        myMSALObj.acquireTokenPopup ("User.Read")
```





Microsoft Graph Permission Names

Azure COMMUNITY CONFERENCE

Pattern: *resource*.*operation*.*constraint*

Constraint determines the potential extent of access the app will have within the directory:

All: grants permission to perform the operations on all resources of the specified type

Shared: grants permission to perform the operations on resources that other users have shared with the signed-in user (mainly used with Outlook resources like mail, calendars, and contacts)

AppFolder: grants permission to read and write files in a dedicated folder in OneDrive (only exposed on Files permissions and only valid for Microsoft accounts)

No constraint: is specified the app is limited to performing the operations on the resources owned by the signed-in user

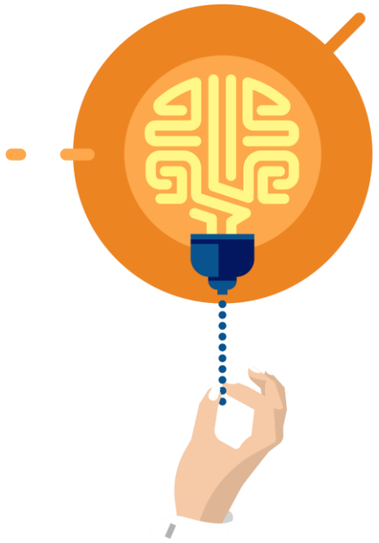
Example: *User*.*Read*.*All*

Permission	Display String	Description	Admin?
User.Read	Sign-in and read user profile	Allows users to sign-in to the app and allows the app to read the profile of signed-in users. It also allows the app to read basic company information of signed-in users.	**No**
User.ReadWrite	Read and write access to user profile	Allows the app to read the signed-in user's full profile. It also allows the app to update the signed-in user's profile information on their behalf.	**No**
User.ReadBasic.All	Read all users' basic profiles	Allows the app to read a basic set of profile properties of other users in your organization on behalf of the signed-in user. This includes display name, first and last name, email address, open extensions and photo. Also allows the app to read the full profile of the signed-in user.	**No**
User.Read.All	Read all users' full profiles	Allows the app to read the full set of profile properties, reports, and managers of other users in your organization, on behalf of the signed-in user.	**Yes**
User.ReadWrite.All	Read and write all users' full profiles	Allows the app to read and write the full set of profile properties, reports, and managers of other users in your organization, on behalf of the signed-in user. Also allows the app to create and delete users as well as reset user passwords on behalf of the signed-in user.	**Yes**

Azure COMMUNITY CONFERENCE

Graph Explorer & quick .NET sample

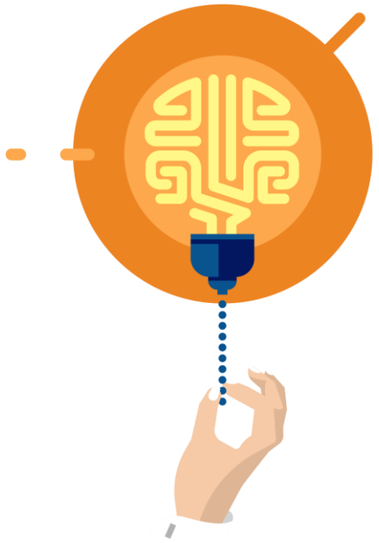
Best Way to learn Graph APIs



#AzConfDev

App Registration

Quintessential for Identity Apps

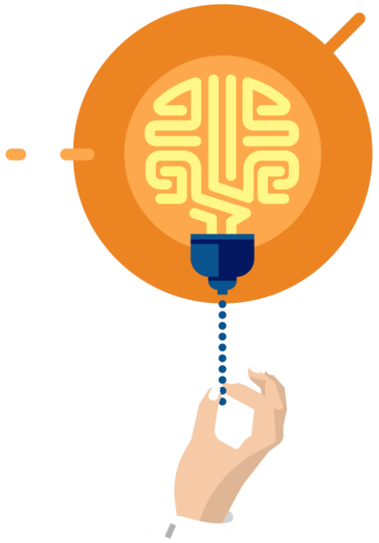


#AzConfDev

Azure COMMUNITY CONFERENCE

Graph SDK & Auth SDK

PRKDemo2



#AzConfDev

Graph API saves time and effort

Language Agnostic

No specific Designs, patterns and Semantics for every system, just unified API
Ex: Identity workload, Teams Workload etc.,

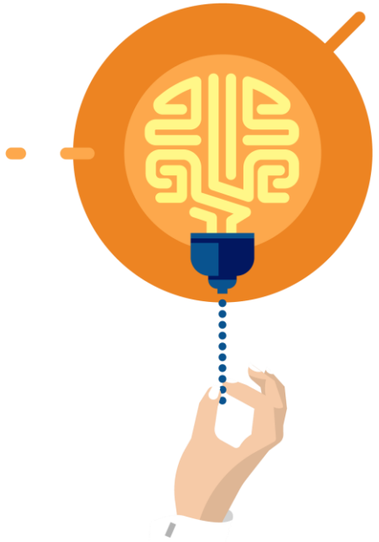
Supports Delta, Pagination and Advanced querying(OData)

Tooling support
Ex: Graph Explorer, Tool Kit, SDKs

Permissions & consent framework is centric to

Advanced Graph Querying

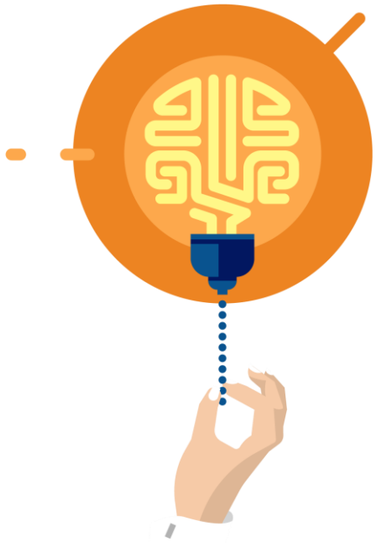
PRKDemo3, PRKDemo4



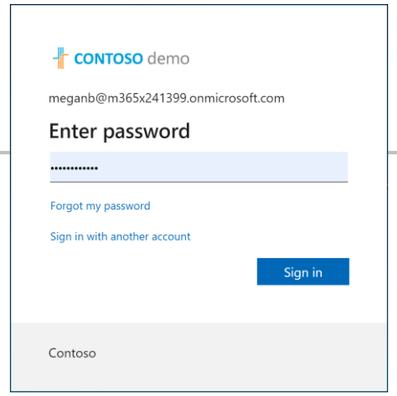
#AzConfDev

Group Membership

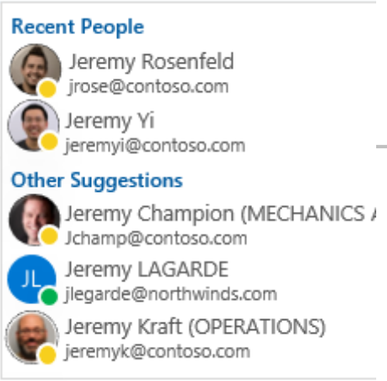
Demo



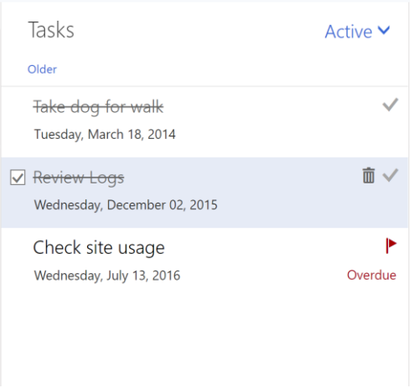
Microsoft Graph Toolkit



login



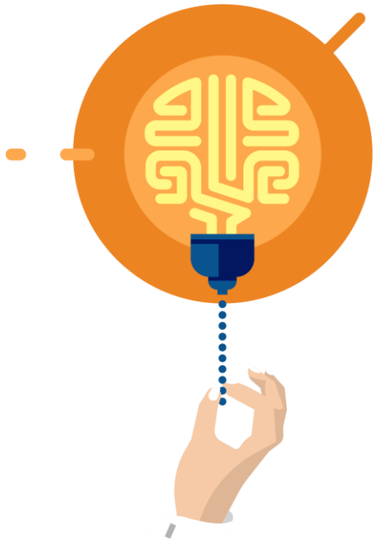
people-picker



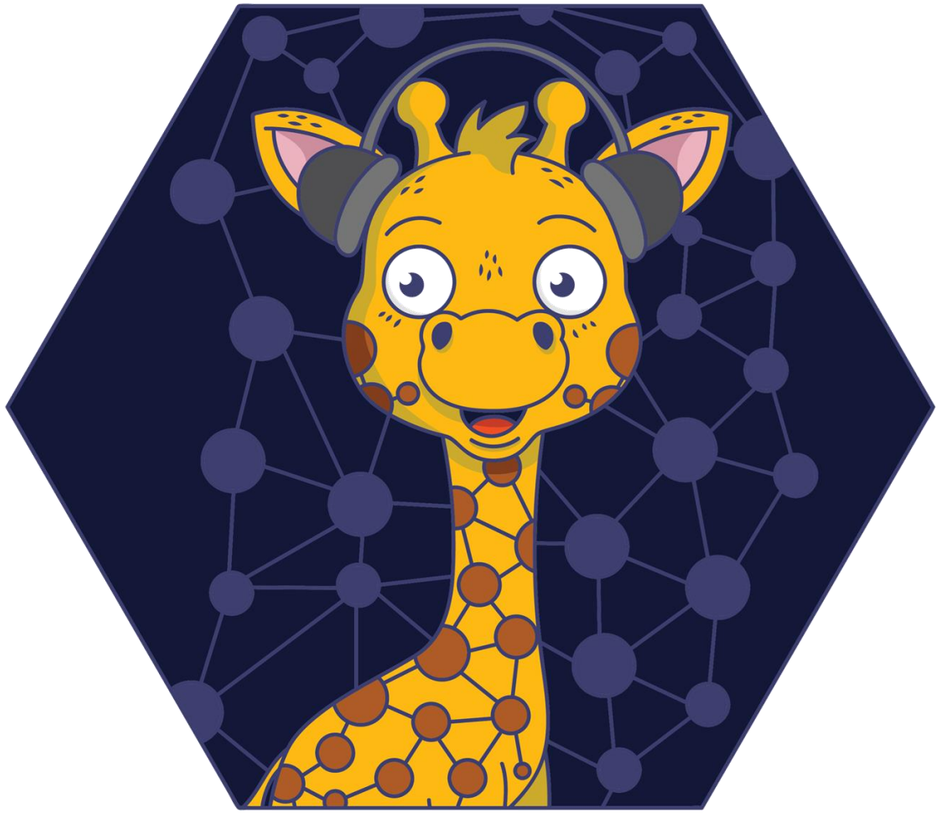
tasks

Quick Demo

Mgt.dev



1. Know the 7 basic operations
2. Learn the 7 basic query parameters
3. Watch for server-side pagination
4. Investigate other query patterns (webhook+delta)
5. Used least privileged permissions



Intent	HTTP METHOD	Description	Example
List	GET	List collection	`GET /users`
Get	GET	Get member of the collection	`GET /users/{id}`
Create	POST/PUT	Create new item in the collection	`POST /users/` `PUT /me/activities/{id}`
Update	PATCH/PUT	Update item	`PATCH /users/{id}` `PUT /me/activities/{id}`
Delete	DELETE	Delete item	`DELETE /users/{id}`
Invoke	POST	Invoke operations	`POST /domains/{id}/verify`
Batch	POST	Execute multiple requests	`POST /$batch`

POST/PATCH/PUT | no response required

If your code doesn’t need to get a response, then opt out

Don’t send unnecessary data over the wire

Tip

Use HTTP **Prefer return=minimal** request header

Some services **always** return 204 No content for PATCH and PUT

Value	Description	Example
`$filter`	Filters results (rows)	`/users?$filter=startsWith(givenName,'J')`
`$select`	Filters properties (columns)	`/users?$select=givenName,surname`
`$orderBy`	Orders results	`/users?$orderBy=displayName desc`
`$top`	Sets the page size of results	`/users?$top=10`
`$expand`	Retrieves related resources	`/groups?$expand=members`
`$count`	Retrieves the total count of matching resources	`/me/messages?$top=2&count=true`
`$search`	Returns results based on search criteria. Currently supported on **messages** and **person** collections	`/me/messages?$search=pizza`

Choose the properties your app really needs and no more

Don’t send unnecessary data over the wire

Tip

Use **$select**

GET https://graph.microsoft.com/v1.0/users?

$select=givenName,mail

Choose the records your app really needs and no more

Don’t send unnecessary data over the wire

Tip

Use **$filter**

GET https://graph.microsoft.com/v1.0/users?

$filter=department eq ‘Sales’ & $select=givenName,mail

Graph uses server-side page size limits

When querying collections, Graph may return the results in many pages

Always expect an `@odata.nextLink` property in the response

Contains the URL to the next page

```
GET https://graph.microsoft.com/v1.0/me/messages?$select=subject,from
```

Response

```
{
    "@odata.nextLink": "https://graph.microsoft.com/v1.0/me/messages?$select=subject%2cfrom&$skip=16",
    "value": [
        {
            "subject": "Your Azure AD Identity Protection Weekly Digest",
```

1.
Always handle the possibility that the responses are paged in nature

2.
Follow the `@odata.nextLink` to obtain the next page of results

3.
Final page will not contain an `@odata.nextLink` property

4.
Treat the entire URL as an opaque string

Scenario

Same scenarios as before, but if you need to optimize further…

Tips

Use **webhook notifications** as the **trigger** to make **delta query** calls

Put notifications in a queue for later processing

Why

Difficult to figure out optimal polling interval

Ex: Query users and then just use the delta query to get the changed users

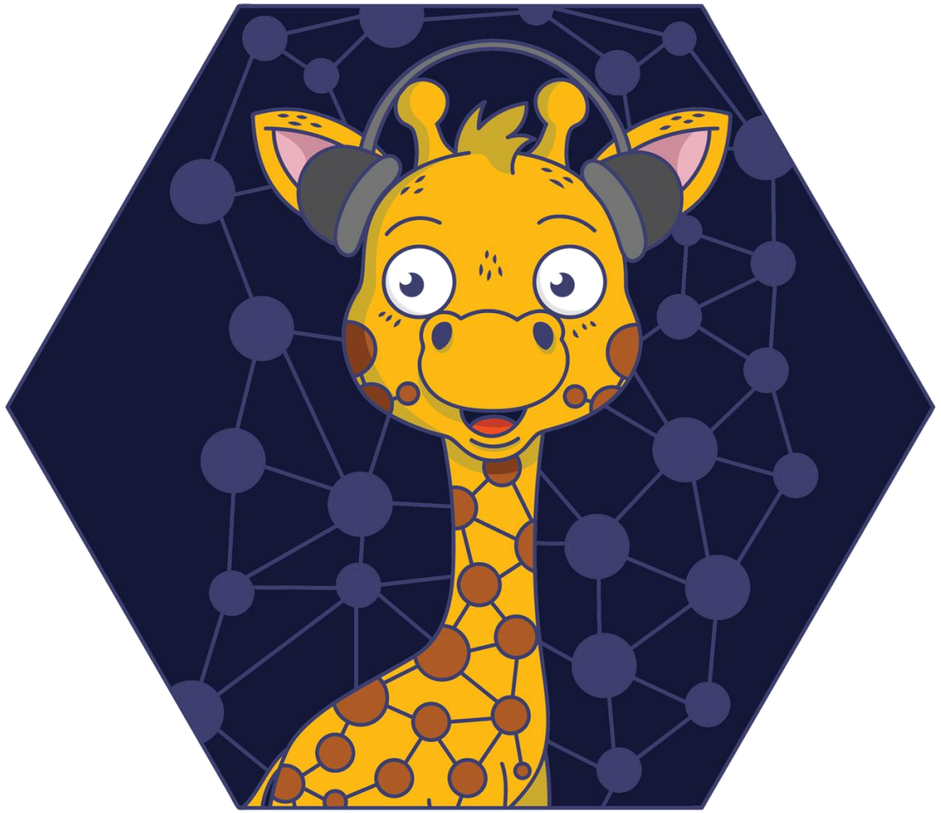
Use least privilege! Only request permissions which are absolutely necessary, and only when you need them

Be thoughtful when configuring your app! This will directly affect end user and admin experiences, along with app adoption and security

When building a multi-tenant app, expect customers to have various application and consent controls in different states

Don’t use AppOnly for user interactive scenarios

1. Know the 7 basic operations
2. Learn the 7 basic query parameters
3. Watch for server-side pagination
4. Investigate other query patterns (webhook+delta)
5. Used least privileged permissions



https://docs.microsoft.com/en-us/graph/migrate-azure-ad-graph-planning-checklist

Step 1: Review the differences between the APIs

Step 2: Examine API use

Step 3: Review app details

Step 4: Deploy, test, and extend your app

Thank you

Kyle Marsh · 1st

Principal Program Manager at Microsoft

Github

Title Sponsors

Microsoft

EY

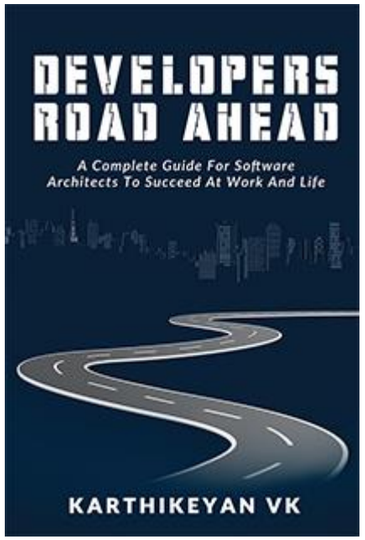
Community Partner

elastic

#AzConfDev

Learning Partners

Packt



Security Partner

FEITIAN
WE BUILD SECURITY

Communities

Azure COMMUNITY CONFERENCE

WOMEN in tech INDIA

TECHMEET360 An initiative by Kovai.co

b.NET

MobConf www.mobconf.com

Chennai Azure User Group

WOMEN WHO CODE DELHI

X hackers

Pehia Foundation

Bengaluru Azure User Group

WOMEN IN CLOUD

PUG Pune User Group

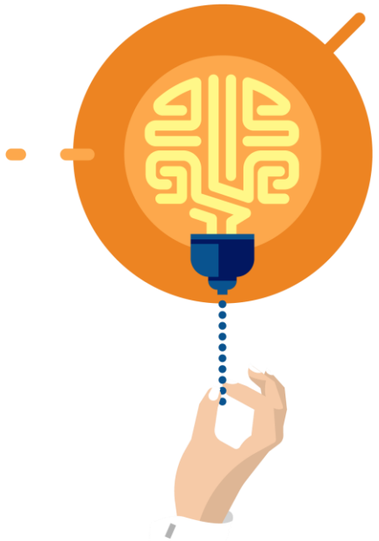
X360 Jump Quietly

Dear Azure

talklify

Azure COMMUNITY CONFERENCE

Q & A



Azure
COMMUNITY
CONFERENCE

Feedback