

Graph API in Power Apps und Automate

Ohne Premium – Lizenzen

PowerSummit 2025



About me



Robert Heep

Microsoft Technical Consultant

Security & Compliance mit Microsoft Purview

Low-Code-Development mit Power Apps / Power Automate

AGENDA

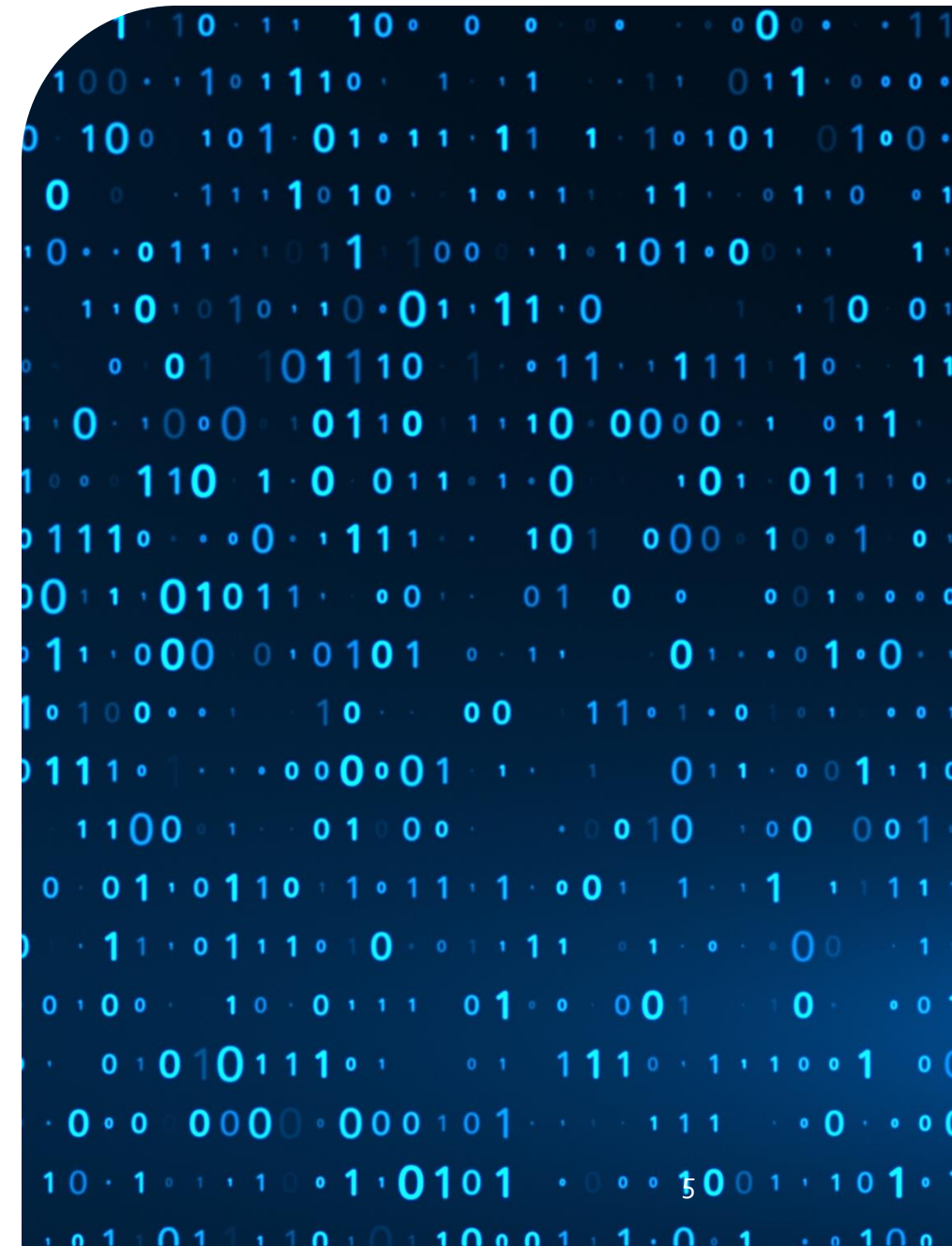


- 01 Was ist die Graph API?
- 02 OOTB Connectors
- 03 HTTP Requests
- 04 Beispiele
- 05 Mystery-Alternative



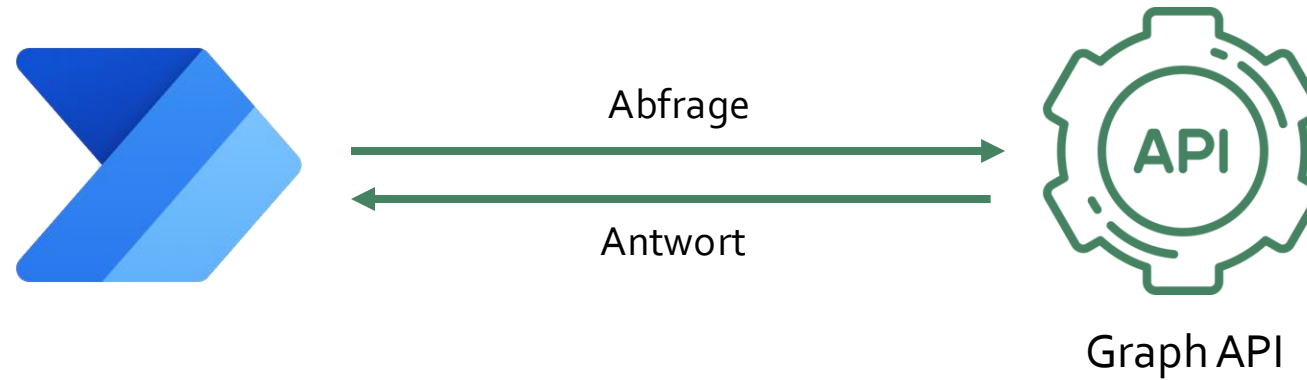
Was ist die Graph API?

Und was kann man damit machen?



Was ist die Graph API?

Schnittstelle zu Daten und Diensten in/aus Microsoft 365



Welche Dokumente hatte ich
zuletzt geöffnet?

Erstelle einen neuen Benutzer

Benutzerinformationen

In welchen Teamräumen bin ich
Mitglied?

Erstelle einen Einladungslink
für ein Dokument

Was ist die Graph API?

Weitere wichtige Begrifflichkeiten

Berechtigungen?

- Application – Berechtigungen unabhängig vom Benutzer (Vorsicht bei Vergabe)
- Delegated – Berechtigungen immer im Kontext des Benutzers



Graph API

Methods (unvollständig)

- GET – Daten lesen
- POST – Daten schreiben

Parameter (unvollständig)

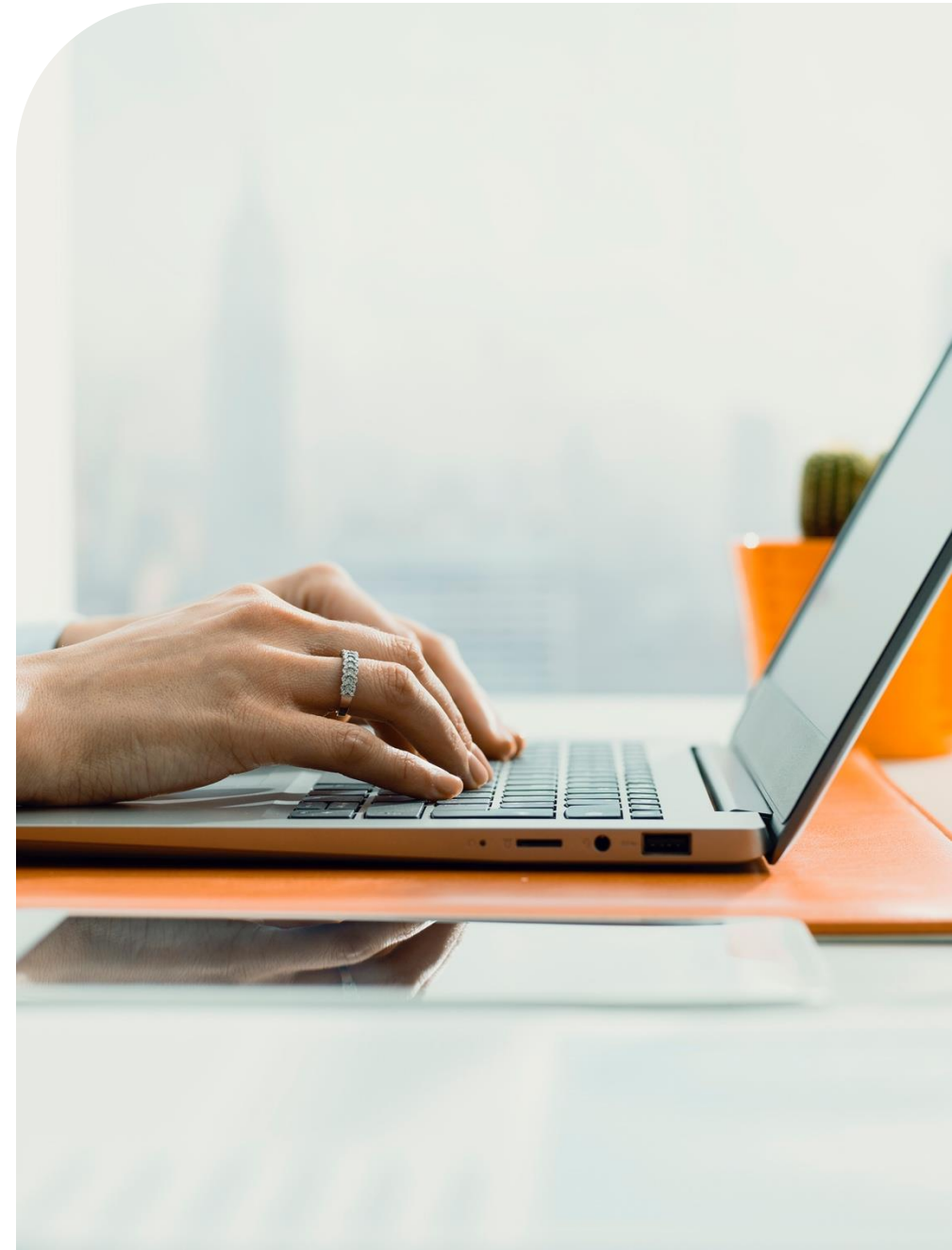
- SELECT – Nur bestimmte Eigenschaften ausgeben
- FILTER – Anfrage filter

Endpoints

- <https://graph.microsoft.com/v1.0/me>
- <https://graph.microsoft.com/v1.0/users>
- <https://graph.microsoft.com/v1.0/groups>

OOTB Aktionen vs. HTTP Requests

Warum reichen die OOTB Aktionen nicht immer aus?



Out-of-the-box (OOTB) connectors

Auszug der Office 365 Connectors

Office 365 Users ☆

- Send an HTTP request
- Get manager (V2)
- Get my trending documents
- Get trending documents
- Get user photo (V2)
- Get direct reports (V2)
- Get my profile (V2)
- Get relevant people

Office 365 Groups ☆

- Add member to group
- Remove member from group
- Update a group event
- List group members
- List groups that I own and belong to
- List my owned groups (V2)

Microsoft Teams ☆

- Add a member to a team
- Create a chat
- Get message details
- List members
- Post a choice of options as the Flow ...
- Post card in a chat or channel
- Create a Teams meeting
- Create a team
- Get messages
- List replies of a channel message
- Post adaptive card and wait for a res...
- Post message in a chat or channel

SharePoint ☆

- Add attachment
- Cancel hub site join request
- Check out file
- Copy folder
- Create file
- Create new document set
- Create sharing link for a file or folder
- Approve hub site
- Check in file
- Copy file
- Create an approval request for an it...
- Create item
- Create new folder
- Discard check out

Office 365 Outlook ☆

- Assign a category to multiple emails
- Contact Management MCP Server
- Create event (V4)
- Email Management MCP Server
- Meeting Management MCP Server
- Send an HTTP request
- Assigns an Outlook category
- Create contact (V2)
- Draft an email message
- Get Outlook category names
- Send a Draft message
- Send an email (V2)

HTTP Requests mit OOTB-Connectors

Jede Office365-Connectorgruppe hat einen HTTP Request

Ausschließlich im **Scope** des jeweiligen **Connectors**

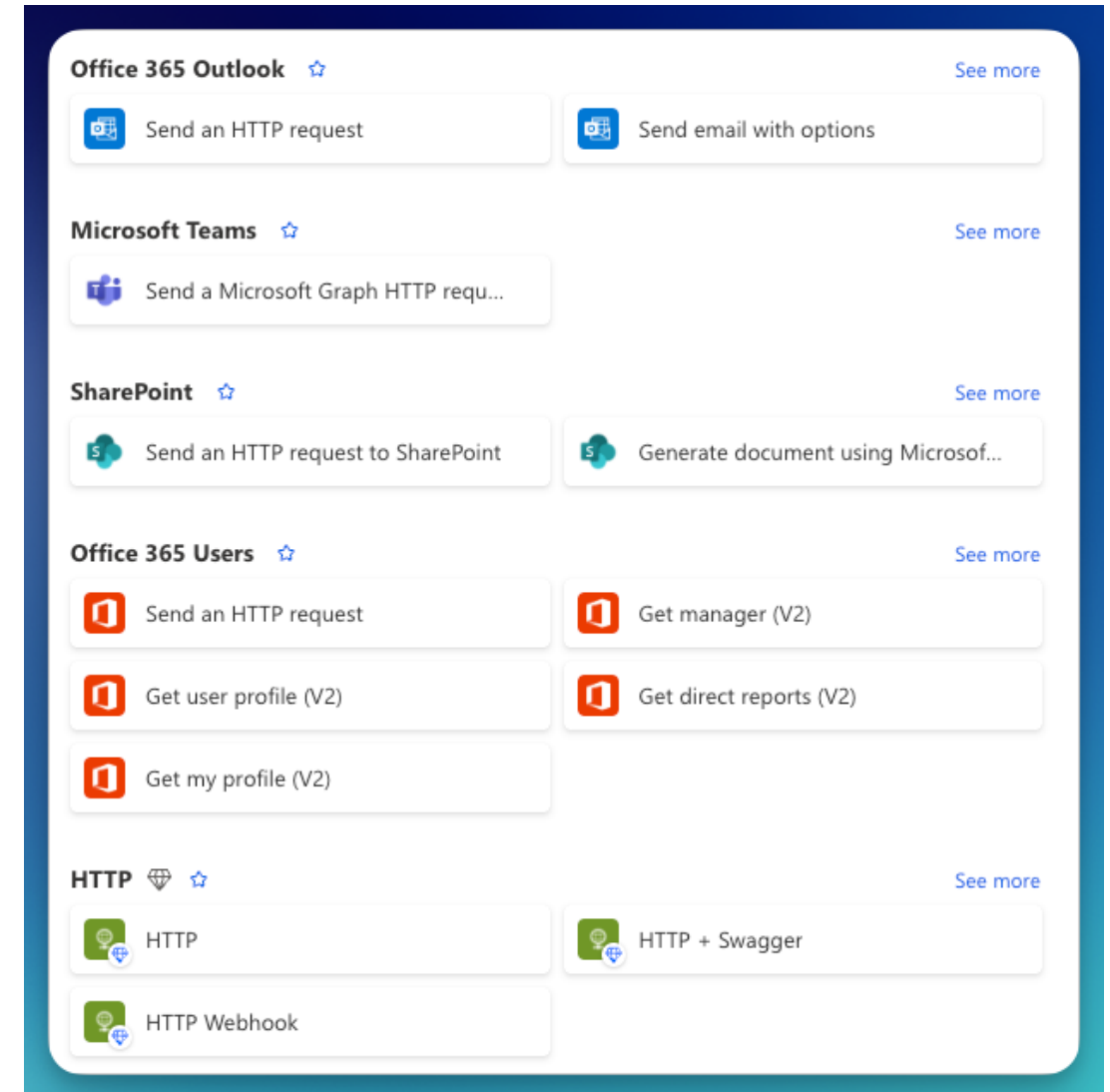
- Bspw. Office 365 User Connector erlaubt
- <https://graph.microsoft.com/v1.0/users>
- <https://graph.microsoft.com/v1.0/me>
- 2nd segment: messages, mailFolders, events, calendar, calendars, outlook, inferenceClassification

IMMER mit **Delegated – Permissions**

SharePoint Connectors besonders, weil

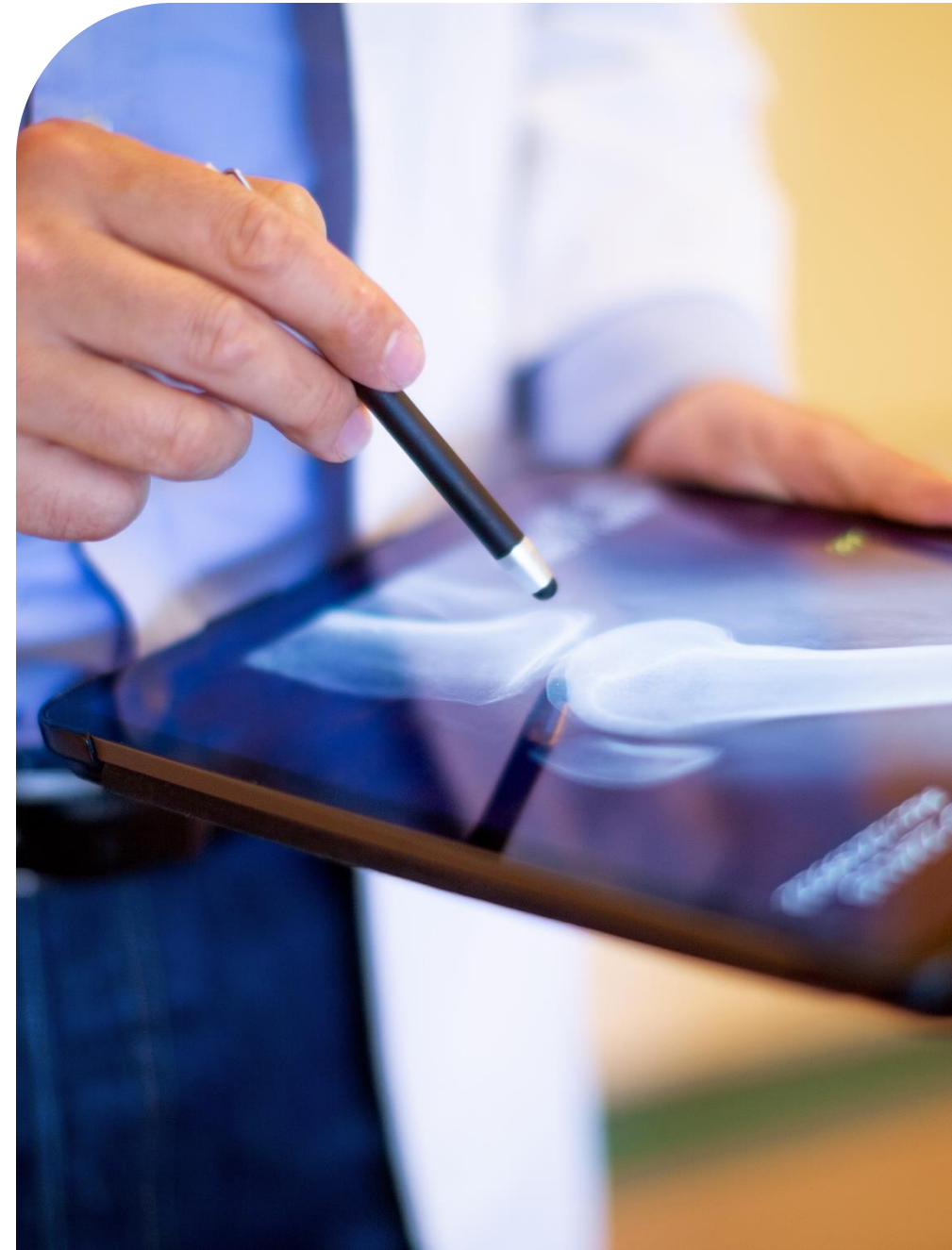
- SharePoint API (als v1)
- Graph API auch möglich (als v2)

Blog: <https://robertheep.de>



Beispiele

Aus der Praxis



Beispiel 1: Groups-Endpoint

Im Beispiel mit \$expand



Szenario:

Ich möchte alle Mitglieder einer Gruppe auslesen, um diese nach Viva Glint zu synchronisieren. Für die Erstellung der Hierarchie benötige ich zusätzlich die Abteilung, Email den Unternehmensnamen des Managers.

Initiale Idee:

Alle Gruppenmitglieder aufzählen, dann mittels Loop die Manager-Daten auslesen und in eine Array-Variable speichern.

Einfacher:

Graph API mit Parameter *expand*, um auf das Manager Property zu erweitern.

The screenshot shows a Power Automate flow titled "Send an HTTP request V2 - get group members and their managers". The flow is in the "Parameters" tab. The URI is set to `https://graph.microsoft.com/v1.0/groups/3dbac2fdb9c7/members/microsoft.graph.user?$expand=manager($select=mail,department,companyName)&$select=displayName,mail,accountEnabled,department,surname,givenName,employeeId,companyName,jobTitle,manager`. The Method is set to GET. The flow is triggered by "Manually trigger a flow". The response is a JSON array of group members, with the first member expanded to show manager details.

```
{
  "displayName": "Hugo Reyes (DE)",
  "mail": "hurley@...",
  "accountEnabled": true,
  "department": null,
  "surname": "Reyes",
  "givenName": "Hugo",
  "employeeId": null,
  "companyName": null,
  "onPremisesSamAccountName": null,
  "jobTitle": null,
  "id": "dee86038-8b33-4744-8467-...",
  "manager": {
    "@odata.type": "#microsoft.graph.user",
    "mail": "admin@...",
    "department": "IT",
    "companyName": "Consulting GmbH"
  }
}
```

Beispiel 2: Teams Beispiel – Nachrichten versenden

Im Beispiel als einfache Teams – Nachricht mit User

Szenario:

Ich möchte viele Teams – Nachrichten in kurzer Zeit verschicken, um Benutzer über den Ausfall eines Systems zu benachrichtigen.

Geht am besten mit:

Graph API und dem Chats-Endpunkt. Kann eine einfache Nachricht sein oder eine Adaptive Card.

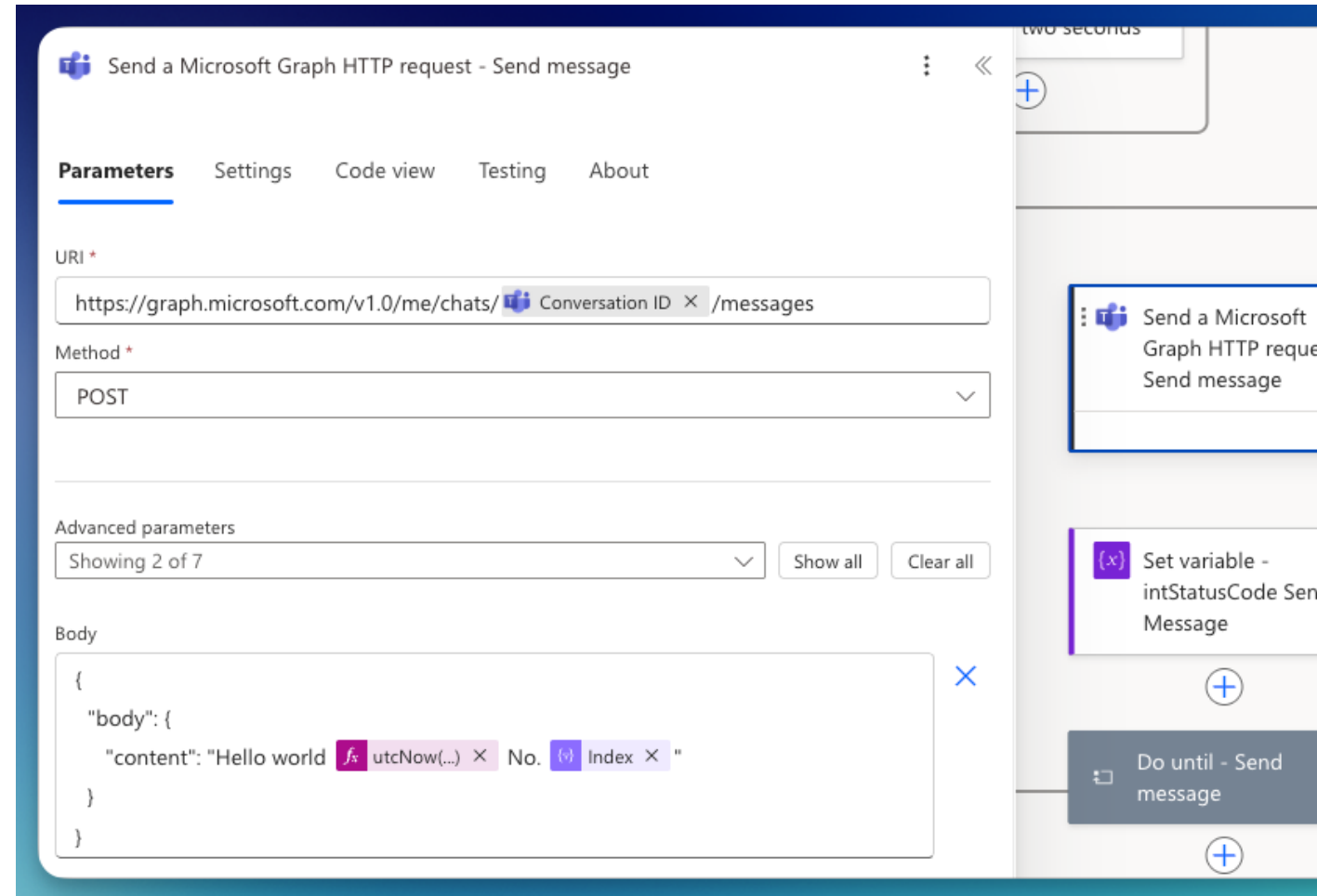
Zum Vergleich:

- Mit Aktion als Flowbot: 25 pro 5 Minuten
- Mit HTTP als User: 20 pro Sekunde

Konkret:

- Mit Aktion 100 Nachrichten: 16min
- Mit HTTP 100 Nachrichten: 5min

Blog: <https://robertheep.de>





Beispiel 3: List group member

Im Beispiel sowohl Direktabfrage als auch Übergabe an Flow

Power Apps kann auch die HTTP Requests nutzen

- Sehr stark eingeschränkt
- Alle Parameter, die weitere Header benötigen (count, search, filter, orderby) funktionieren nicht
- Nur bis 999 Elemente (mehr -> besser mit Power Automate)

```
Set(
    grpMembers,
    Office365Groups.HttpRequestV2(
        "https://graph.microsoft.com/v1.0/groups/" & ComboboxCanvas1.Selected.id & "/members/microsoft.graph.user?$expand=manager($select=mail,department,companyName)&$select=displayName,mail,accountEnabled,department,surname,givenName,employeeId,companyName,onPremisesSamAccountName,jobTitle,manager",
        "GET",
        "",
        {ContentType: "application/json"}
    );
ClearCollect(
    colMembers,
    ForAll(
        Table(grpMembers.value),
        {
            DisplayName: Text(Value.displayName),
            Company: Text(Value.companyName),
            Department: Text(Value.department),
            Mail: Text(Value.mail),
            Id: Text(Value.id),
            ManagerMail: Text(Value.manager.mail),
            ManagerDepartment: Text(Value.manager.department),
            ManagerCompany: Text(Value.manager.companyName)
        }
    )
);
```

Beispiel 4 (ohne Demo): SharePoint Online

Ein absolutes Muss!

SharePoint API extrem wichtig. Bspw. ist möglich:

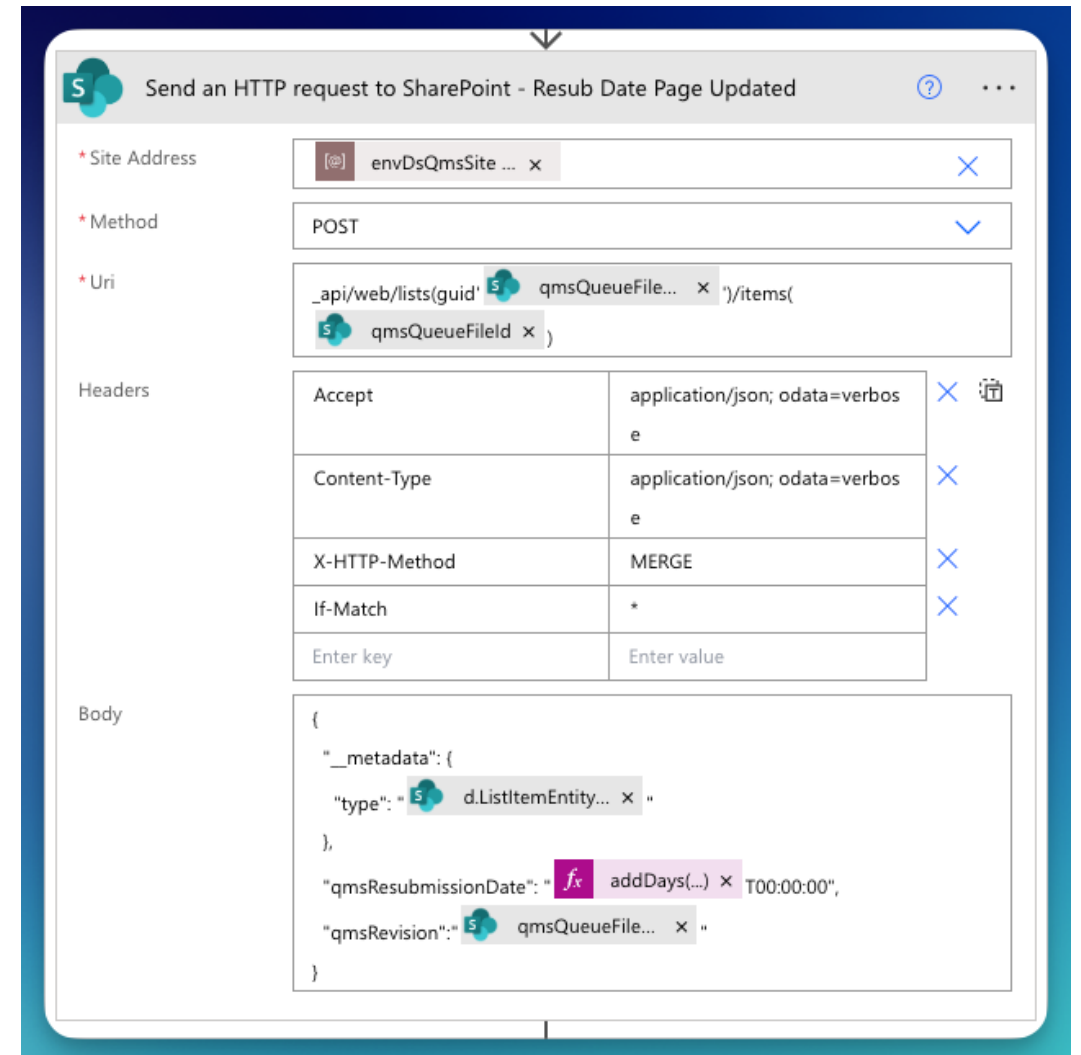
- Elemente aktualisieren ohne Pflichtdaten angeben zu müssen
- Batch – Anlage von Elementen
- Listen und Bibliotheken erstellen

Mehrere Versionen (und damit auch Graph API möglich)

- V1 = `_api/web/lists`
- V2 = `_api/v2.o/drives`

Mehr zu v2: <https://learn.microsoft.com/en-us/sharepoint/dev/apis/sharepoint-rest-graph>

Blog: <https://robertheep.de>



The screenshot shows a REST client interface with the following configuration:

- Title:** Send an HTTP request to SharePoint - Resub Date Page Updated
- Site Address:** envDsQmsSite ...
- Method:** POST
- Uri:** `_api/web/lists(guid' qmsQueueFile...')/items( qmsQueueFileId )`
- Headers:**

Accept	application/json; odata=verbose
Content-Type	application/json; odata=verbose
X-HTTP-Method	MERGE
If-Match	*
Enter key	Enter value
- Body:**

```
{
  "__metadata": {
    "type": "d.ListItemEntity..."
  },
  "qmsResubmissionDate": "addDays(...) T00:00:00",
  "qmsRevision": " qmsQueueFile..."
}
```

Warum also HTTP Requests nutzen?

Hat gleich mehrere Vorteile



Höhere **Performance**



Mehr **Flexibilität**



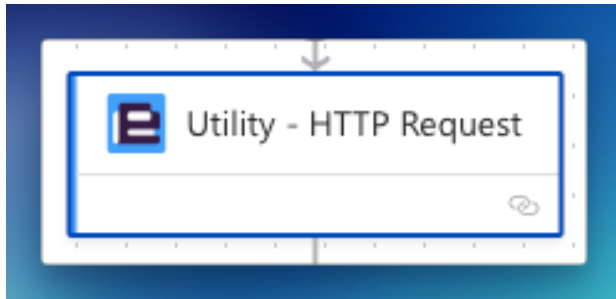
Automatisch tieferes **Wissen**

Die Mystery - Alternative

Encodian Flowr

Encodian hat einen eigenen HTTP Connector gebaut

- 1:1 Abbildung des Microsoft eigenen HTTP
- Kann Raw, Basic und Entra-Authentifizierung



Vorteile

- 0.05 credits per operation (bei Standard-Paket heißt das bis zu 10.000 Anfragen pro Monat)
- Achtung! Beim kostenlosen Plan entspricht eine Ausführung 1 credit!
- Nicht pro Benutzer lizenziert

Blog: <https://robertheep.de>



Wo soll ich anfangen?

Quellen und Nachschlagewerk

Mein Blog 🐼

<https://robertheep.de/all-blogs/http-requests-using-built-in-actions-power-apps-and-automate>

<https://robertheep.de/all-blogs/http-request-to-sharepoint-demystified>

SharePoint API Explorer, für das Finden von Endpunkten

<https://s-kainet.github.io/sp-rest-explorer/#/entity/SP.List>

Offizielle Microsoft-Quellen

<https://developer.microsoft.com/en-us/graph/graph-explorer>

<https://learn.microsoft.com/en-us/graph/api/overview?view=graph-rest-1.0&preserve-view=true>

<https://learn.microsoft.com/en-us/sharepoint/dev/apis/sharepoint-rest-graph>

<https://learn.microsoft.com/en-us/sharepoint/dev/sp-add-ins/get-to-know-the-sharepoint-rest-service?tabs=csom>

Q&A / Socials



Robert Heep

Blog <https://robertheep.de>

LinkedIn <https://www.linkedin.com/in/robertheep/>