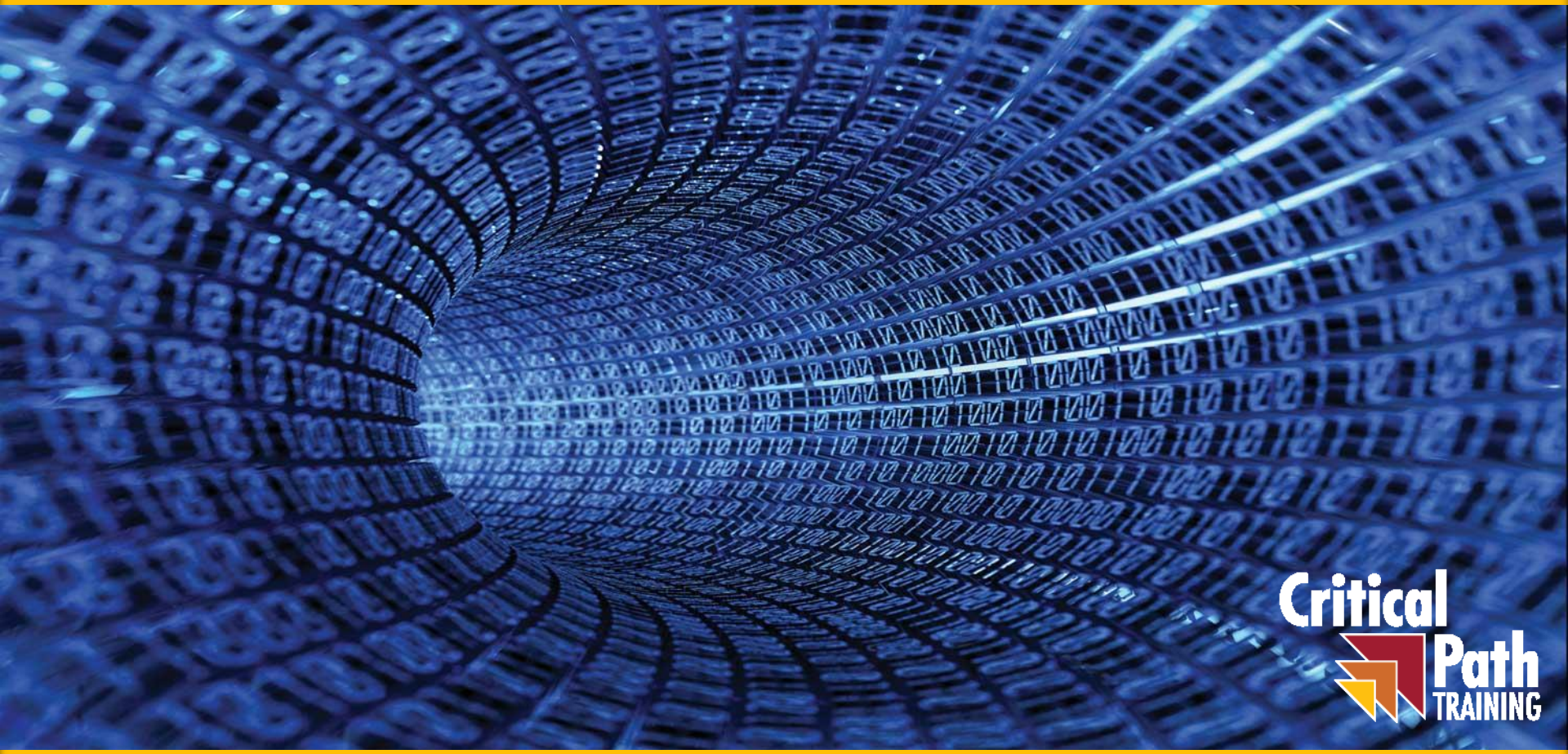
Programming with the Microsoft Graph API

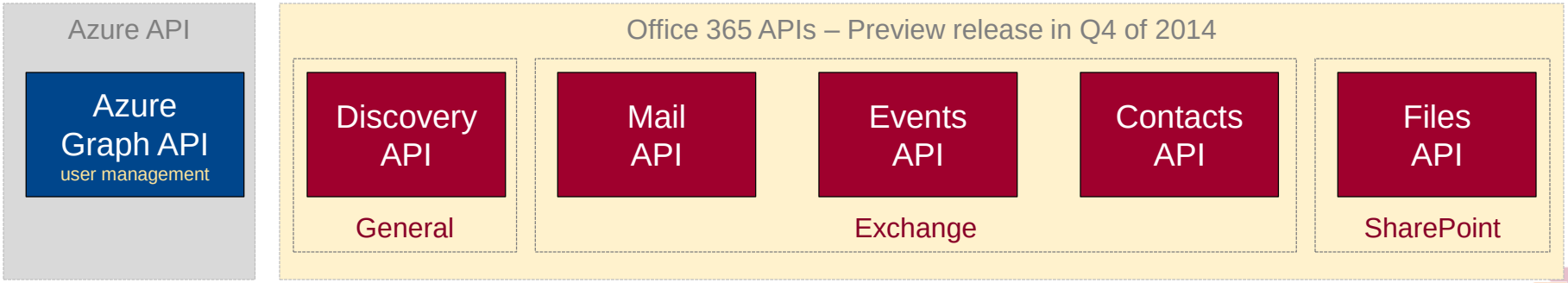


Agenda

- Overview of Microsoft Graph API
- Constructing URLs for the Microsoft Graph API
- Developing Applications with the Microsoft Graph API
- Programming SPFx Webparts using MSGraphClient
- Granting Microsoft Graph API Permissions

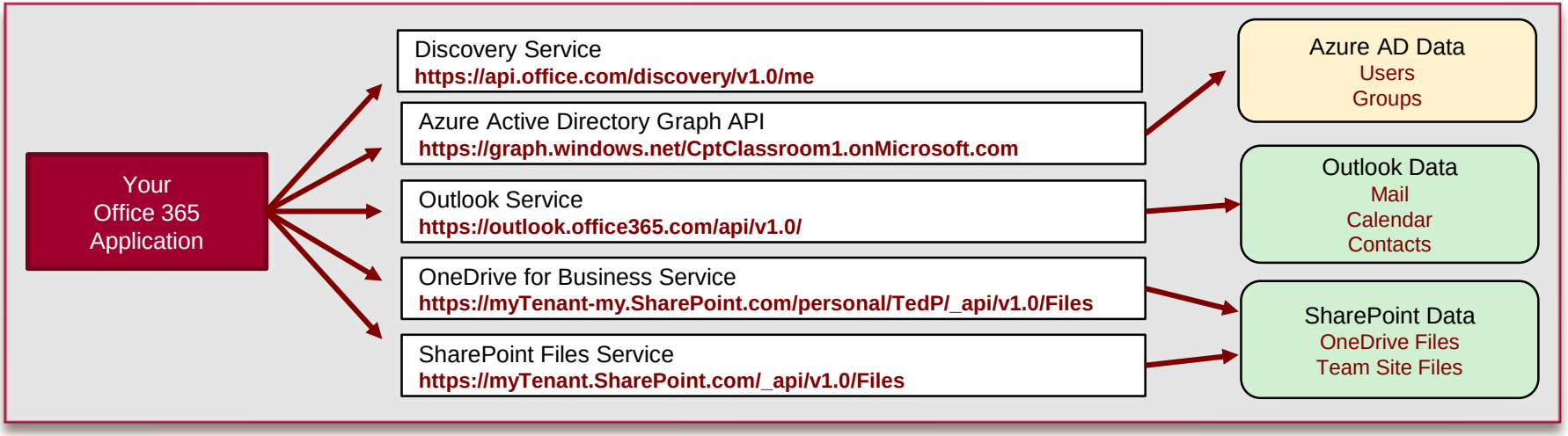
Microsoft Released Office 365 APIs in 2014

- Office 365 APIs created for accessing data in Office 365
  - Implemented as RESTful services based on ODATA version 4.0
  - Provides authentication and authorization based on OAuth 2.0
  - Provides extra authentication support for OpenID Connect
- Open standards provide wide range of accessibility
  - Many choices for tools, languages and development platforms
  - Microsoft has created Office 365 SDKs for specific platforms
- Office 365 APIs set the stage for the Microsoft Graph API
  - Microsoft Graph API created to simplify accessing Office 365



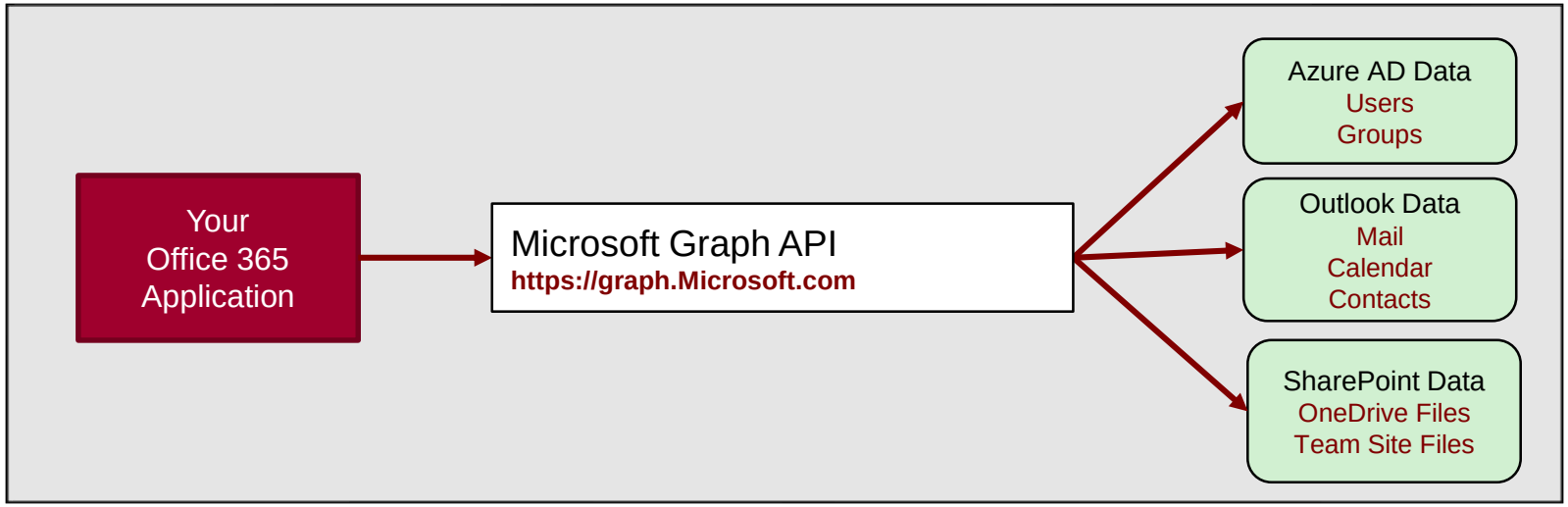
Office 365 API Endpoint in Initial Release

- Azure Endpoints
  - Azure Graph API
- Office 365 API Endpoints
  - Outlook service
  - OneDrive for Business Service
  - SharePoint Files Service
  - Discovery Services



The Microsoft Graph API

- Designed as a single, more-comprehensive service
  - Abstracts away divisions between AD, Exchange and SharePoint
  - No need to discover endpoints using the Discovery Service
  - You can acquire and cache a single access token per user



What Does It Do?

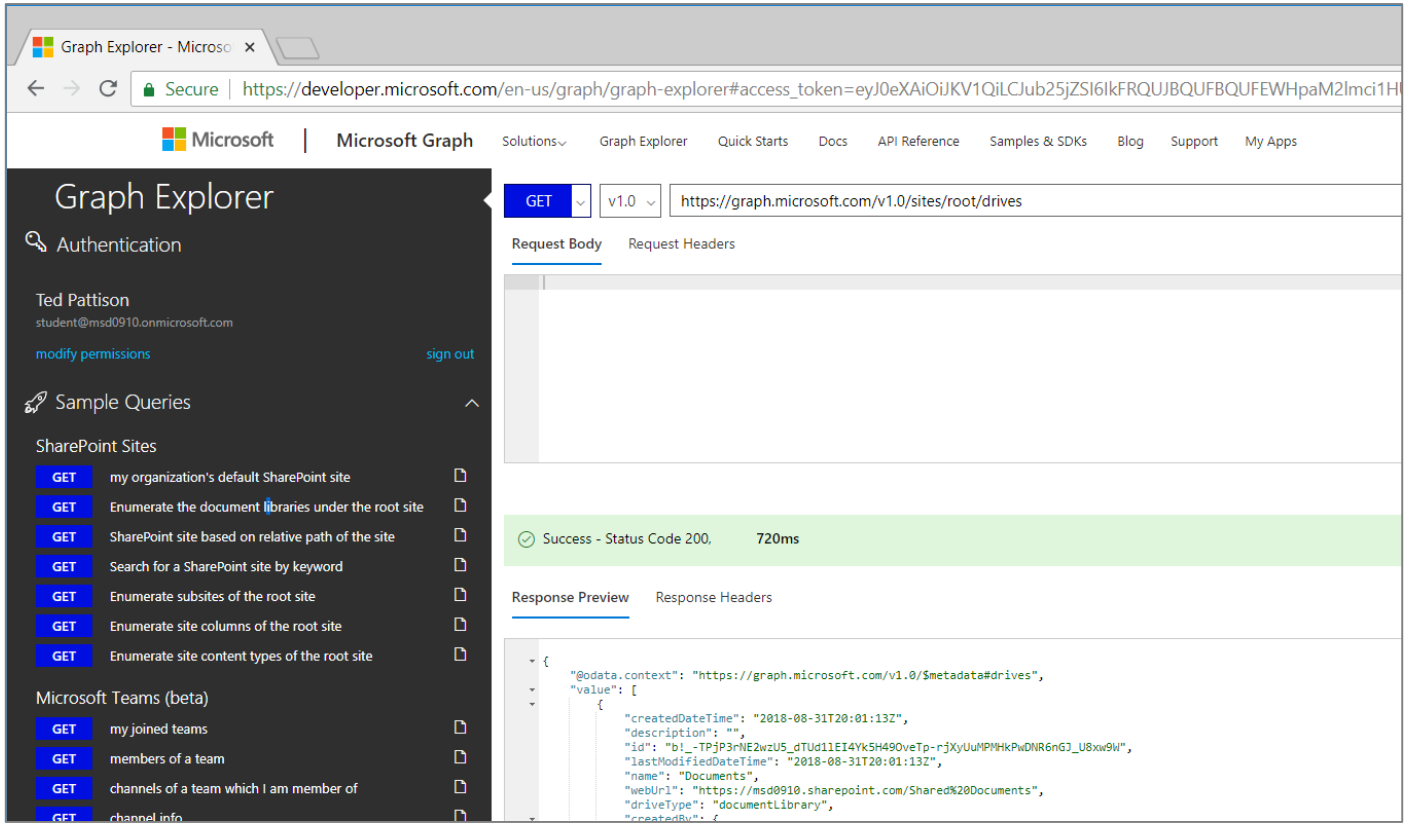
- Single resource that proxies multiple Microsoft services
- Allows for easy traversal of objects and relationships
- Simplifies token acquisition and management
- Eliminates the need to traditional discovery
  - It does this by using "me" and "myorganization"

Universal Access

- Direct REST API
  - Any platform
  - Any language
  - Any framework
- Native SDKs
  - Utilize framework & platform specific implementations
  - Abstracts the details of building & processing requests over HTTP
  - .NET, iOS, Android, PhP, Ruby, JavaScript, etc.

The Graph Explorer

- Graph Explorer used to execute calls into Microsoft Graph API
  - Great place to discover what he API can do
  - https://developer.microsoft.com/en-us/graph/graph-explorer



Agenda

- ✓ Overview of Microsoft Graph API
- ➢ Constructing URLs for the Microsoft Graph API
- Developing Applications with the Microsoft Graph API
- Programming SPFx Webparts using MSGraphClient
- Granting Microsoft Graph API Permissions

Calling the Graph API

```
https://graph.microsoft.com
/{version}/{resource}/{id}/{property}?{query-parameters}
```

- HTTP verbs dictate the request intent: GET | POST | PATCH | PUT | DELETE
- Version: `/v1.0` or `/beta`
- Resource: `/users`, `/groups`, `/sites`, `/drives`, `/devices`, more…
- Member from collection: `/users/AAA`
- Property: `/users/AAA/department`
- Traverse to related resources via navigations: `/users/AAA/events`
- Query parameters: `/users/AAA/events?$top=5`
  - Format results: `$select` | `$orderby`
  - Control results: `$filter` | `$expand`
  - Paging: `$top` | `$skip` | `$skiptoken`

Pagination

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

Request

```
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

Querying data | Use filters

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

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

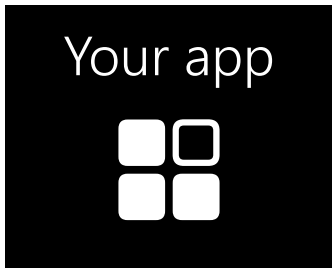
Tip

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

- User, group and organizational

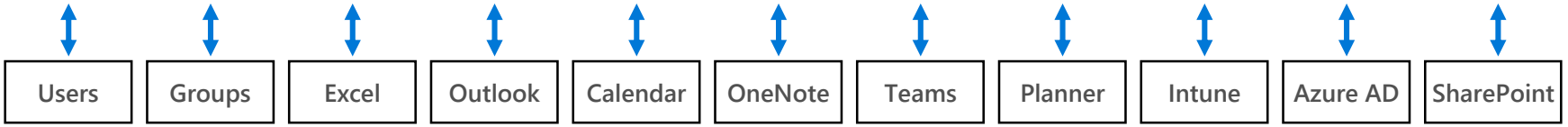
Data

One endpoint
One token
All users



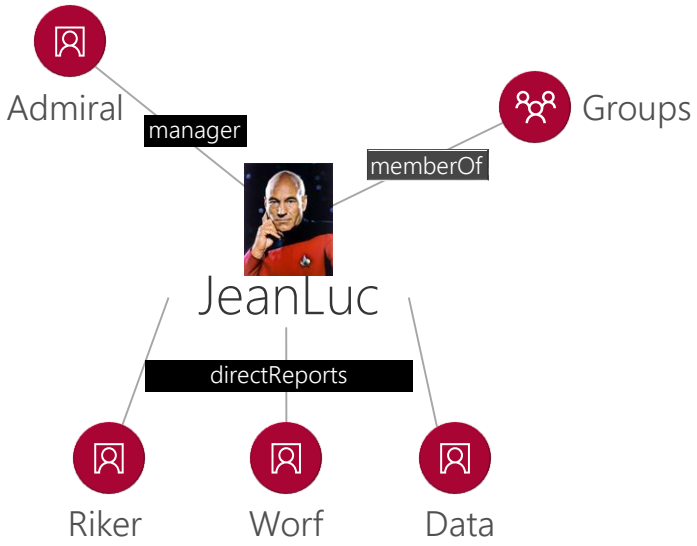
Microsoft Teams API - preview
Project Rome API - preview
SharePoint Sites API – GA
OneNote API – GA
Planner API – GA

https://graph.microsoft.com



With Microsoft Graph

Get the user profile




```
GET: /users/JeanLuc
{
  "displayName": "JeanLuc Picard",
  "jobTitle": “Captain",
}
```

```
GET: /users/JeanLuc/photo/…
{}
```

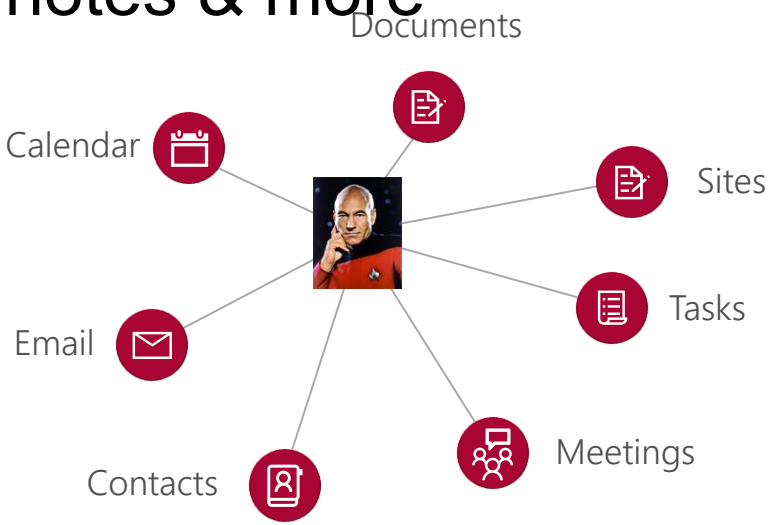
```
GET: /users/JeanLuc/manager
{"displayName": "Starfleet Admiral", …}
```

```
GET: /users/JeanLuc/directReports
"value" : [
  {"displayName": "Riker", …},
  {"displayName": "Worf", …},
]
```

```
GET: /me/memberOf/…
"value" : [
  {"displayName": "Starfleet engineering", …},
  {"displayName": "Beaming", …},
]
```

With Microsoft Graph

Get content for email, calendar, files, tasks, sites, notes & more




```
GET /me/drive/root/…
"value" : [
  {"name": "proposal.pptx",… },
  {"name": "forecast.xlsx",… }
]
```

```
GET /drives/items/{id}/workbook
```

```
GET /me/messages
```

```
GET /me/events
```

```
GET /me/contacts
```

```
GET /me/onenote/notebooks
```

```
GET /me/planner/tasks
```

```
GET /me/devices
```

```
GET /sites:/teams/opg:/
```

```
GET /sites:/teams/opg:/lists
```

```
GET /groups/{id}/conversations
```

With Microsoft Graph

Get insights based on activities


```
GET /me/insights/trending
"value" : [
  {"name": "presentation.pptx", …},
  {"name": "forecast.xlsx", …}
]

GET /me/drive/recent
"value" : [
  {"name": "guidelines.pptx", …},
  {"name": "budget.xlsx", …}
]

GET people/?$search="topic: planning"
"value" : [
  {"displayName": "Data", …},
  {"displayName": "Worf", …},
]

POST: /me/findMeetingTimes
{
  "attendees": [
    {
      "type": "required",
      "emailAddress": {
        "address": "riker@contoso.com"
      }
  ],
  "meetingDuration": "2h"
}
```

Read data from existing SharePoint site

GET https://graph.microsoft.com/beta/sites/{siteId}/lists/{listId}/items?expand=columnSet


```
{ "value":[ {
    "createdBy": { "user" : {"id":"d54e4cdd-d2ca-4c39-bfa5-35895bca12f0","displayName":"John"}},
    "createdDateTime":"2017-09-20T08:16:21Z",
    "eTag":"1610ac6a-24f6-4458-9733-1e5977c63caa,1",
    "id":"1610ac6a-24f6-4458-9733-1e5977c63caa",
    "lastModifiedBy":{"user":{"id":"d54e4cdd-d2ca-4c39-bfa5-35895bca12f0","displayName":"John"}},
    "lastModifiedDateTime":"2017-09-0T08:16:21Z",
    "webUrl":"https://site.sharepoint.com/sites/mysite/Lists/mytasks/1_.000",
    "listItemId":1,
    "columnSet":{
      "Title":"Project Upgrade: Use the Microsoft Graph",
      "Description":"Set up group for new technologies.",
      "id":"1",
      ...
 } }]}
```

OneDrive + Excel Services

```
GET https://graph.microsoft.com/v1.0/me/drive/
            root/search(q='.xlsx')?select=name,id,webUrl
```

```
GET https://graph.microsoft.com/1.0/me/drive/
      items/<id>/workbook/worksheets
```

```
GET https://graph.microsoft.com/beta/me/drive/
            items/{itemId}/workbook/worksheets('Time')/
            range(address='a2:d4')
```

Update Excel Timesheet Data

```
PATCH https://graph.microsoft.com/beta/me/drive/
      items/{itemId}/workbook/worksheets('Sheet1')/
      range(address='a2:b2')
{
  "values": [ ["September", "200.0"] ],
  "valueTypes": [ ["String", "Double"] ],
}
```

New Capabilities

- Traversal of relationships
- Query parameters
- Batching - preview
- Notifications - users & groups - preview
- Track changes - GA
- Extensions - GA

Agenda

- ✓ Overview of Microsoft Graph API
- ✓ Constructing URLs for the Microsoft Graph API
- ➢ Developing Applications with the Microsoft Graph API
- Programming SPFx Webparts using MSGraphClient
- Granting Microsoft Graph API Permissions

Microsoft Graph JavaScript SDK

- Microsoft Graph is accessible via REST API & various SDKs

- Client-side solutions can leverage the JavaScript SDK
  - https://github.com/microsoftgraph/msgraph-sdk-javascript
  - Requires initialization with an Azure AD provided OAuth2 access token to create the client

Initializing the Microsoft Graph JS SDK

```
var client = MicrosoftGraph.Client.init({
  authProvider: (done) => {
    done(null, access_token);
  }
});

client
  .api('/me')
  .get((err, res) => {
    console.log(res);
  });
```

Microsoft Graph TypeScript Type Declarations

- Use the Microsoft Graph JavaScript SDK in TypeScript applications
- TypeScript type declarations introduce strong types and documentation to client-side projects
  - https://github.com/microsoftgraph/msgraph-typescript-typings

Microsoft Graph TypeScript Type Declarations

```
import * as MicrosoftGraph from '@microsoft/microsoft-graph-types';

// init Microsoft Graph client

client
  .api('/me')
  .get((error: any, user: MicrosoftGraph.User, rawResponse?: any) => {
    console.log('name: ', user.displayName);
    console.log('email: ', user.displmailayName);
    console.log('phone: ', user.businessPhones[0]);
    });
  });
```

Agenda

- ✓ Overview of Microsoft Graph API
- ✓ Constructing URLs for the Microsoft Graph API
- ✓ Developing Applications with the Microsoft Graph API
- ➢ Programming SPFx Webparts using MSGraphClient
- • Granting Microsoft Graph API Permissions

SharePoint Online User Already Authenticated

- Users in SharePoint Online & Office 365 are already authenticated
  - SharePoint Online has same Azure AD dependency as Microsoft Graph
  - Users login to Office 365 with their Work & School account (Azure AD)

- Calls to the Microsoft Graph are proxied through SharePoint Online
  - Eliminates the need for creation of a separate Azure AD application
  - Does not bypass any permission / scope requirements
  - Can only access business entities, not consumer entities

Calling Services from Client-side Code

- Calling a secured service requires the acquisition of token
  - Authenticate and get authorized in Azure AD application
  - Interactive login & programmatic acquisition of access token

- Client-side components cannot do the same
  - Components cannot securely do this across domains seamlessly
  - Can’t store application ID’s and secrets client side
  - Require an authentication prompt with a popup or redirection

SPFx Includes Microsoft Graph Client

- MSGraphClient: Microsoft Graph Client fpr SPFx
  - Abstracts the token acquisition from the SPFx development
  - Wraps the Microsoft Graph JavaScript SDK line

```
let graphClient: MSGraphClient =
    this.context.serviceScope.consume(MSGraphClient.serviceKey);
```

SPFx Proxy Calls through Existing Application

- SharePoint Online already has an Azure AD application
  - Client-side solutions in SharePoint Online call the SharePoint REST API in the same domain
  - No extra authentication required
  - Provided these tenants have granted the necessary scopes, SharePoint will call the Microsoft Graph
  - Responses from the Microsoft Graph are returned back to the client-side application
- Permission requests to Azure AD applications
  - Only SharePoint Online tenant administrators can [grant|reject] permission requests
  - Approved permissions are available to all client-side solutions in a tenant

SPFx Solutions Declare Permission Requests

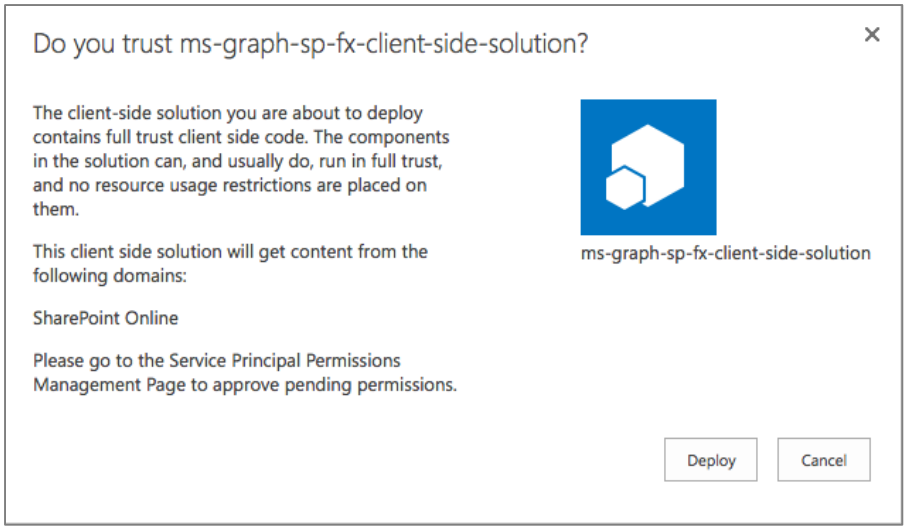
```
// package-solution.json
{
  "$schema": "https://developer.microsoft.com/json-schemas/spfx-build/package-solution.schema.json",
  "solution": {
    "name": "ms-graph-sp-fx-client-side-solution",
    "id": "dfb230b7-4f61-431f-9b65-a34e83922663",
    "version": "1.0.0.0",
    "includeClientSideAssets": true,
    "webApiPermissionRequests": [
      { "resource": "Microsoft Graph", "scope": "User.ReadBasic.All” },
      { "resource": "Microsoft Graph", "scope": "Calendars.Read” },
      { "resource": "Microsoft Graph", "scope": "Tasks.Read” }
    ]
  },
  "paths": {
    "zippedPackage": "solution/ms-graph-sp-fx.sppkg"
  }
}
```

Agenda

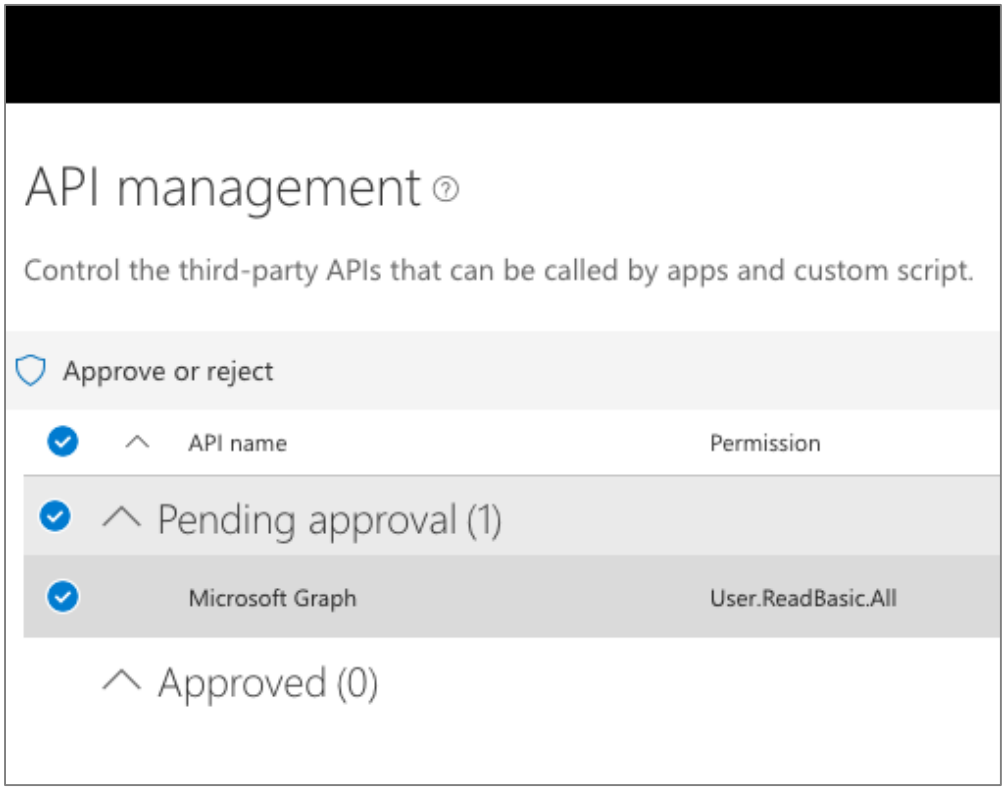
- ✓ Overview of Microsoft Graph API
- ✓ Constructing URLs for the Microsoft Graph API
- ✓ Developing Applications with the Microsoft Graph API
- ✓ Programming SPFx Webparts using MSGraphClient
- ➢ Granting Microsoft Graph API Permissions

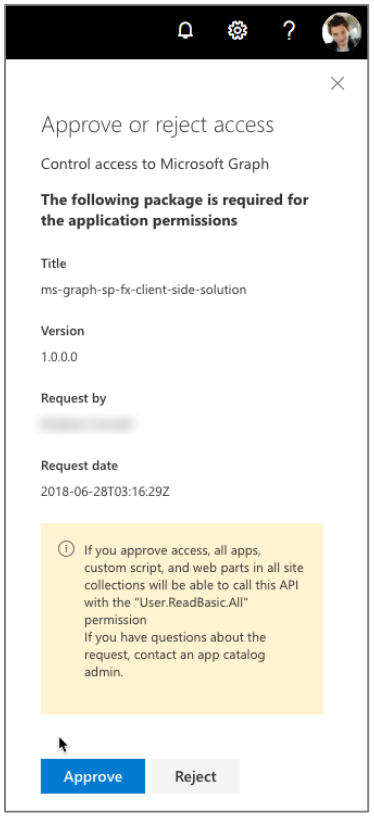
Add Package to SharePoint App Catalog

- Extra note in dialog notifies of additional step required
- While application can be installed in SharePoint sites, it does not have the permissions granted that it needs to access Azure AD protected resources



Approve / Reject with SharePoint Online API Management Page



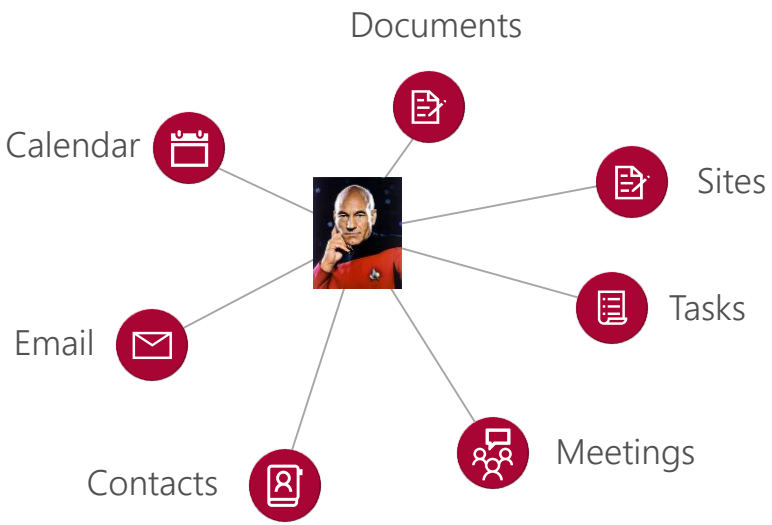


Summary

- ✓ Overview of Microsoft Graph API
- ✓ Constructing URLs for the Microsoft Graph API
- ✓ Developing Applications with the Microsoft Graph API
- ✓ Programming SPFx Webparts using MSGraphClient
- ✓ Granting Microsoft Graph API Permissions

With Microsoft Graph

Use $batch to combine multiple requests in one call

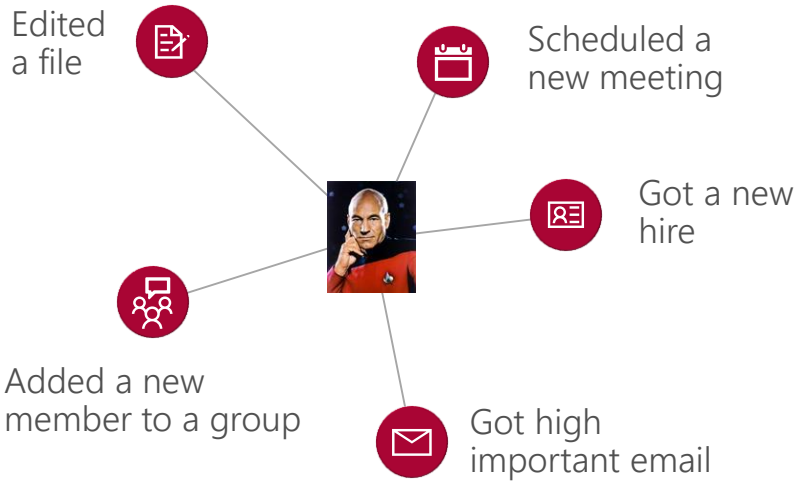


Batching

```
POST /$batch
{
  "requests": [{
     "id": "1",
     "url": "/me/drive/root/children",
     "method": "POST",
     "body": {
         "name": "folder1",
         "folder": {}
     },
     "headers": {
         "content-type": "application/json"
     }
   }, {
      "id": "2",
      "url": "/me/drive/root/children/folder1",
      "method": "GET",
      "dependsOn": ["1"]
   }, {
      "id": "3",
      "method": "GET",
      "url": "/me/planner/tasks"
   }, {
      "id": "4",
      "method": "GET",
      "url": "/groups/{id}/events"
   }
 ]
}
```

With Microsoft Graph

Get notifications & track changes



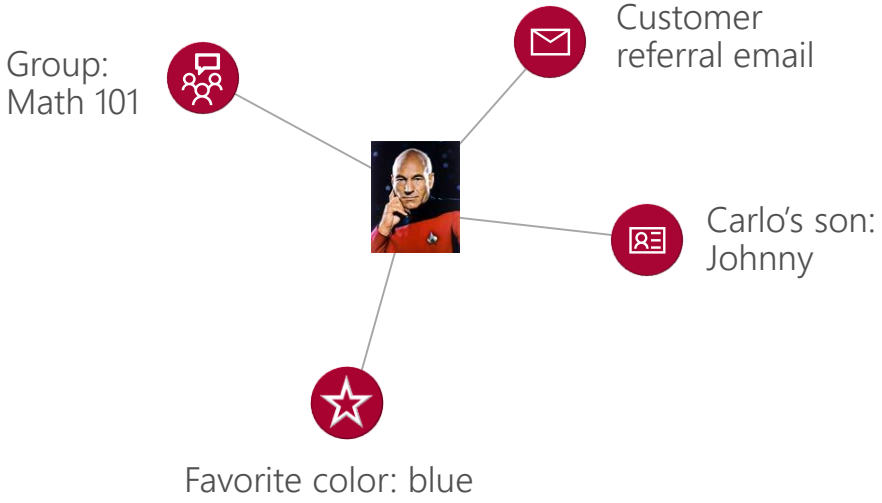

```
GET/me/mailFolders/{id}/messages/delta
"@odata.deltalink":"me/mailfolders('AA')/messages/
delta?$deltatoken=BB",
"value" : […]
```

```
POST /subscriptions
{
  "changeType": "created,updated",
  "notificationUrl": "https://app.net/callback",
  "resource": "/me/mailfolders('AA')/messages",
}
```

```
GET/me/mailFolders/{id}/messages/delta
?$deltatoken=BB"
"value" : […]
```

Extending the Graph

Add extensions to user, group, mail & more



Open Extensions

```
GET /me/message/<id>/?$expand=extensions
{
 "displayName": "JeanLuc",
 "extensions": [
    {
      "extensionName": "Com.Contoso.Referral",
      "companyName": "atwork",
      "expirationDate": "2017-12-30T11",
      "dealValue": 10,000
    }
  ]
}
```

Schema extensions

```
POST /schemaExtensions
{
    "id": "training_courses",
    "targetTypes": [ "Group" ],
    "properties": [
        {
            "name": "courseName",
            "type": "String"
        }…
    ]
}
```

```
GET /groups?$filter=courses/name eq Math101
```