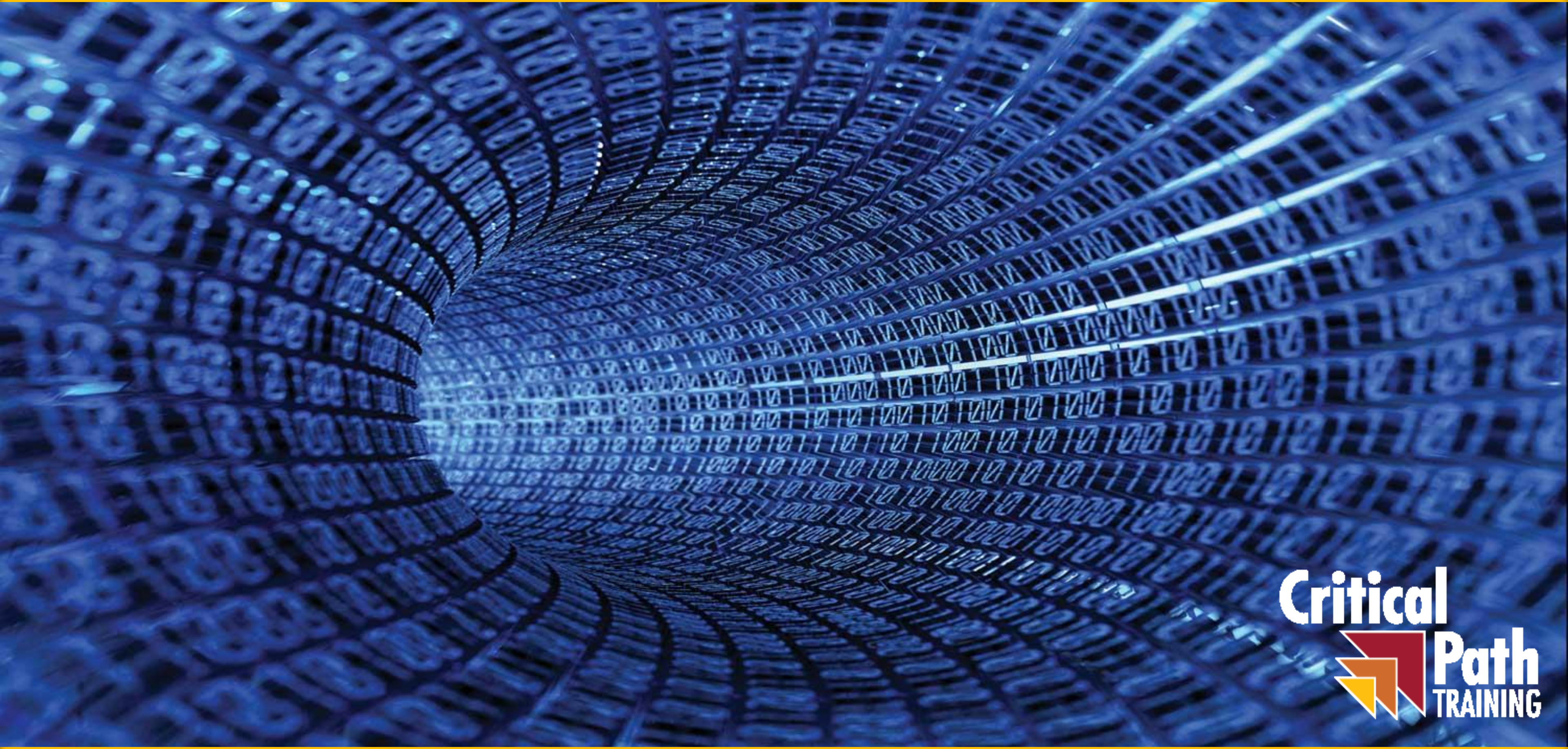
Developing with the Microsoft Graph API



Agenda

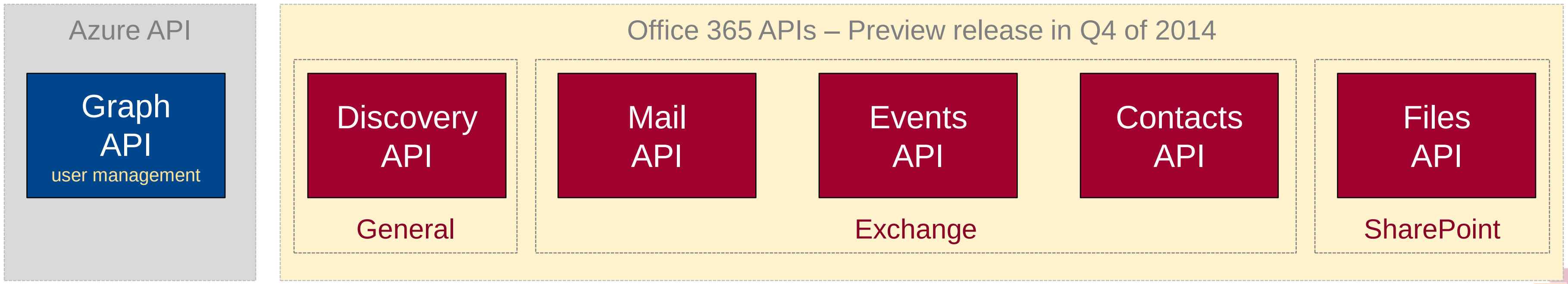
- Overview of the Microsoft Graph API
- Developing with the Microsoft Graph API
- Creating Users and Groups in Office 365
- Programming Messages and Calendar Events
- Managing Files and Documents in OneDrive

Evolution of the SharePoint/Office Platform

- Web Part Packages
  with the release of SharePoint 2003
- Farm Solutions (aka Full Trust Solutions)
  with the release of SharePoint 2007
- Sandbox Solutions
  with the release of SharePoint 2010
- SharePoint ~~App~~ Add-in Model
  with the release of SharePoint 2013
- Applications built using the Office 365 APIs
  released in preview by Microsoft in October of 2014

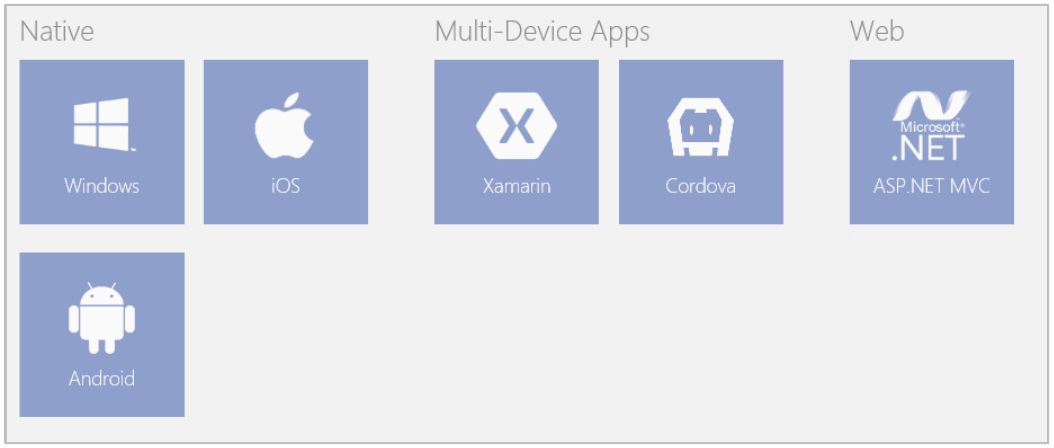
Office 365 API Overview

- New web service APIs 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 less complex than Exchange and SharePoint APIs
  - No need to become a “career programmer"



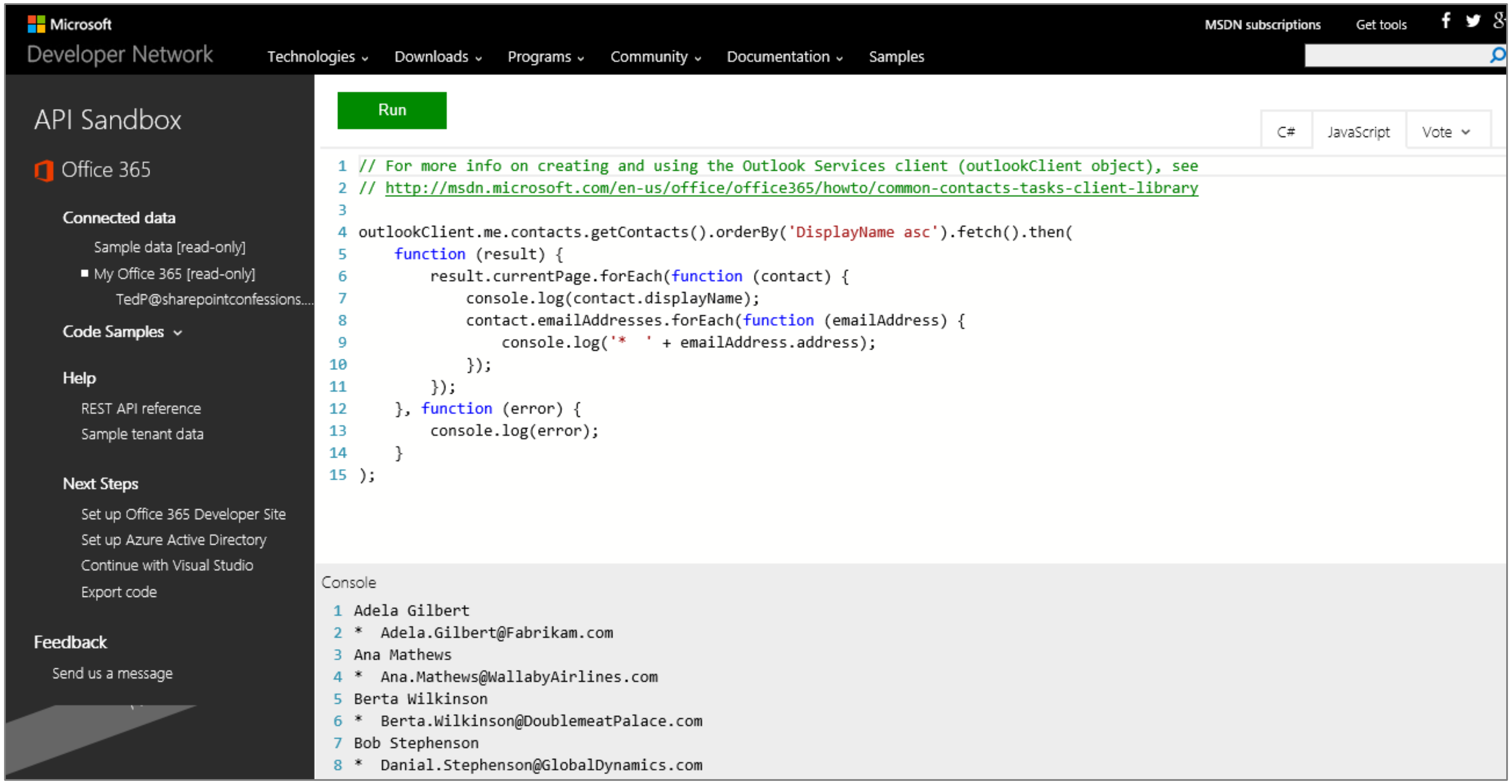
Support for Non-Microsoft Platforms

- Office 365 APIs universally accessible via open standards
  - Any development platform can use REST, ODATA and OAuth2
  - Office 365 developers can use wide array of tools and languages
  - No hosting environment dependencies (e.g. Windows or IIS)
- SDKs and extra help available for selected platforms
  - SDKs available for Windows, ASP.NET, iOS and Android
  - Choose between Visual Studio, XCode, Eclipse or Android Studio



Office 365 API Sandbox

- Simple online utility provided by MSDN
  - https://apisandbox.msdn.microsoft.com/



```
// For more info on creating and using the Outlook Services client (outlookClient object), see
// http://msdn.microsoft.com/en-us/office/office365/howto/common-contacts-tasks-client-library

outlookClient.me.contacts.getContacts().orderBy('DisplayName asc').fetch().then(
    function (result) {
        result.currentPage.forEach(function (contact) {
            console.log(contact.displayName);
            contact.emailAddresses.forEach(function (emailAddress) {
                console.log('*   ' + emailAddress.address);
            });
        });
    }, function (error) {
        console.log(error);
    }
);
```

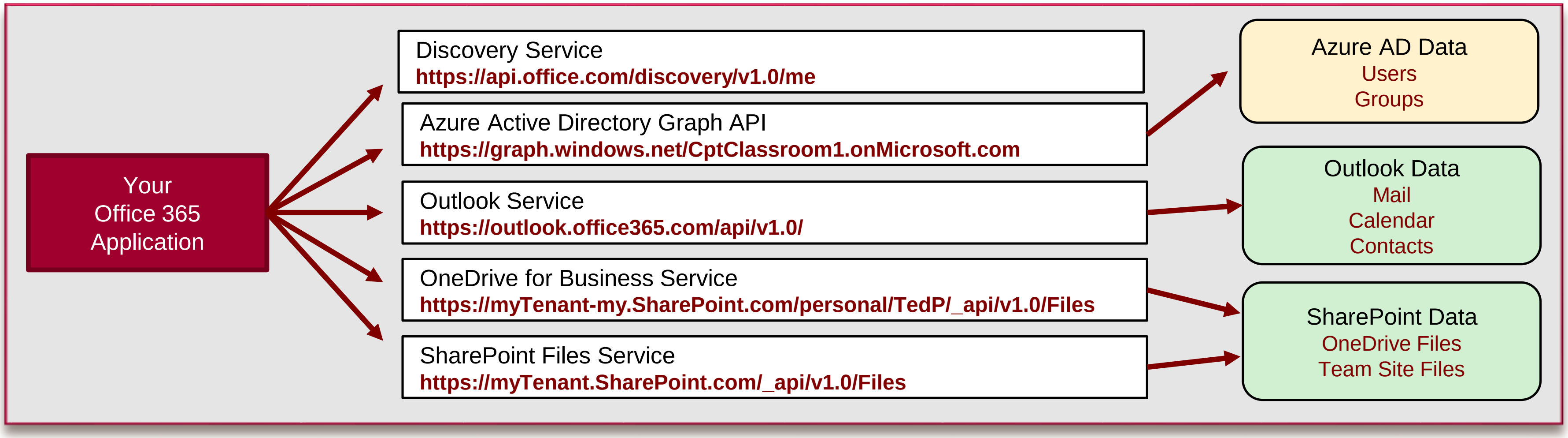
```
Adela Gilbert
*  Adela.Gilbert@Fabrikam.com
Ana Mathews
*  Ana.Mathews@WallabyAirlines.com
Berta Wilkinson
*  Berta.Wilkinson@DoublemeatPalace.com
Bob Stephenson
*  Danial.Stephenson@GlobalDynamics.com
```

Agenda

- ✓ Introduction to the Office 365 APIs
- ➢ Office 365 API Service Endpoints
- Understanding the Microsoft Graph API
- Programming the Microsoft Graph API

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

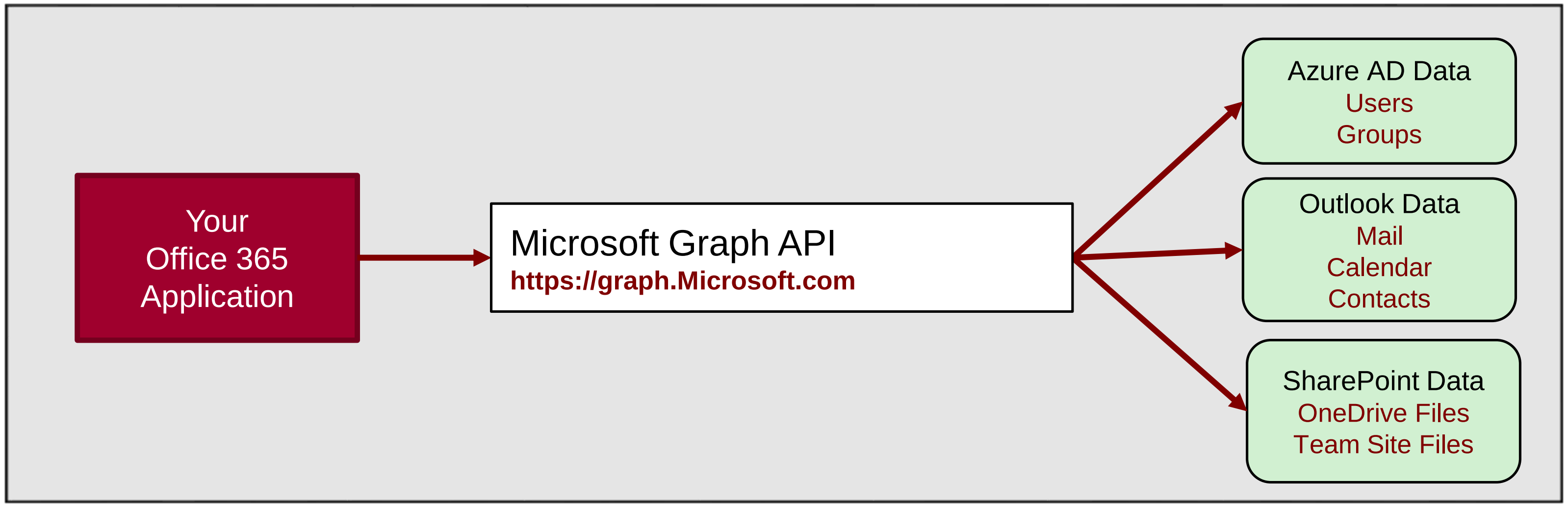

Your Office 365 Application
Discovery Service
https://api.office.com/discovery/v1.0/me
Azure Active Directory Graph API
https://graph.windows.net/CptClassroom1.onMicrosoft.com
Outlook Service
https://outlook.office365.com/api/v1.0/
OneDrive for Business Service
https://myTenant-my.SharePoint.com/personal/TedP/_api/v1.0/Files
SharePoint Files Service
https://myTenant.SharePoint.com/_api/v1.0/Files
Azure AD Data
Users
Groups
Outlook Data
Mail
Calendar
Contacts
SharePoint Data
OneDrive Files
Team Site Files

Agenda

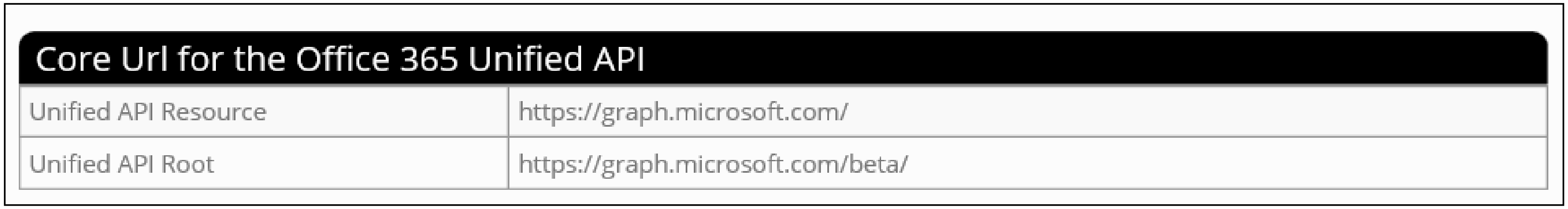
- ✓ Introduction to the Office 365 APIs
- ✓ Office 365 API Service Endpoints
- ➢ Understanding the Microsoft Graph API
- Programming the Microsoft Graph API

The Microsoft Graph API

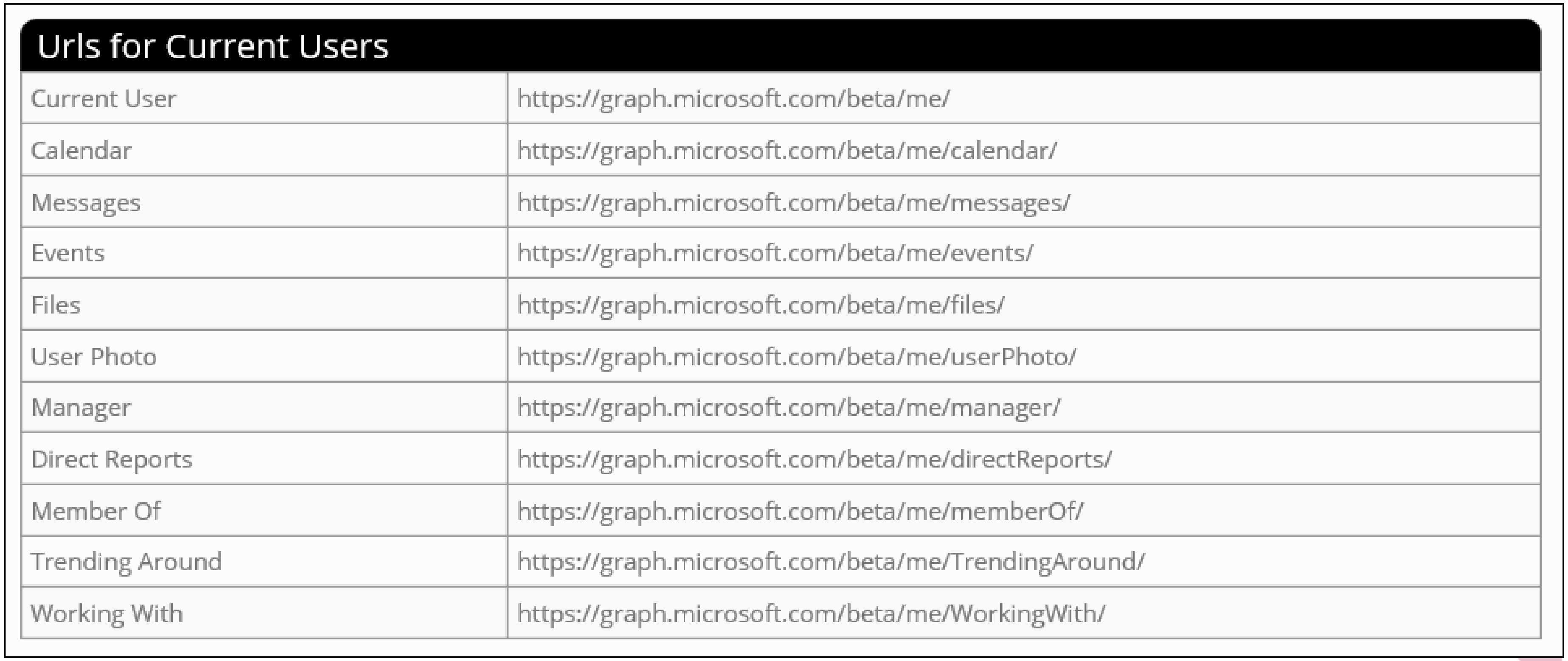
- Designed as a single, more-comprehensive service
  - Abstracts away divisions between AD, Exchange and SharePoint
  - No need to discover endpoints using the Discovery Service
  - You can acquire and cache a single access token per user



Unified API URLs

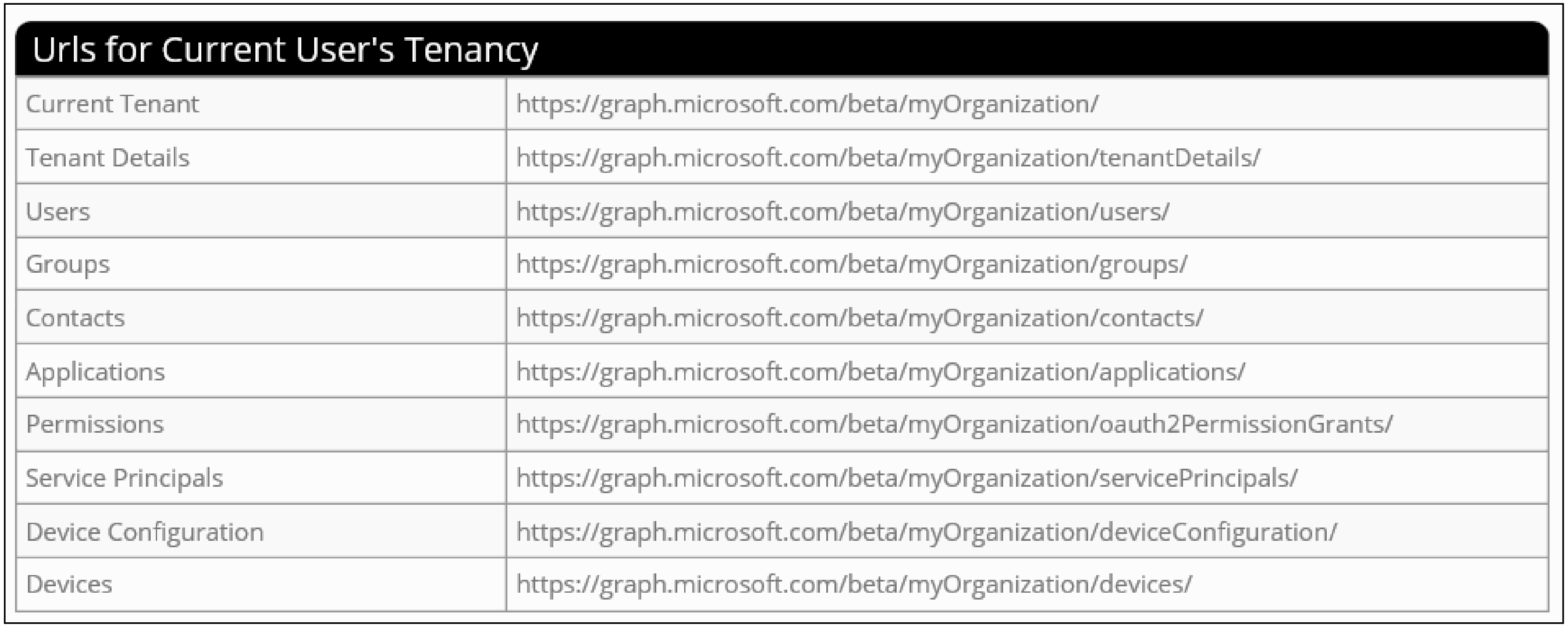


Core Url for the Office 365 Unified API	
Unified API Resource	https://graph.microsoft.com/
Unified API Root	https://graph.microsoft.com/beta/



Urls for Current Users	
Current User	https://graph.microsoft.com/beta/me/
Calendar	https://graph.microsoft.com/beta/me/calendar/
Messages	https://graph.microsoft.com/beta/me/messages/
Events	https://graph.microsoft.com/beta/me/events/
Files	https://graph.microsoft.com/beta/me/files/
User Photo	https://graph.microsoft.com/beta/me/userPhoto/
Manager	https://graph.microsoft.com/beta/me/manager/
Direct Reports	https://graph.microsoft.com/beta/me/directReports/
Member Of	https://graph.microsoft.com/beta/me/memberOf/
Trending Around	https://graph.microsoft.com/beta/me/TrendingAround/
Working With	https://graph.microsoft.com/beta/me/WorkingWith/

More Unified API URLs

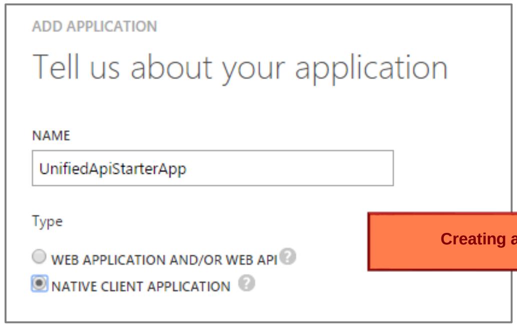


Urls for Current User's Tenancy

Current Tenant	https://graph.microsoft.com/beta/myOrganization/
Tenant Details	https://graph.microsoft.com/beta/myOrganization/tenantDetails/
Users	https://graph.microsoft.com/beta/myOrganization/users/
Groups	https://graph.microsoft.com/beta/myOrganization/groups/
Contacts	https://graph.microsoft.com/beta/myOrganization/contacts/
Applications	https://graph.microsoft.com/beta/myOrganization/applications/
Permissions	https://graph.microsoft.com/beta/myOrganization/oauth2PermissionGrants/
Service Principals	https://graph.microsoft.com/beta/myOrganization/servicePrincipals/
Device Configuration	https://graph.microsoft.com/beta/myOrganization/deviceConfiguration/
Devices	https://graph.microsoft.com/beta/myOrganization/devices/

Creating an Azure AD Application

- To access the Office 365 APIs…
  - application must first be registered with Azure Active Directory
  - Application registered as Web Application or as Native Client


ADD APPLICATION

Tell us about your application

NAME

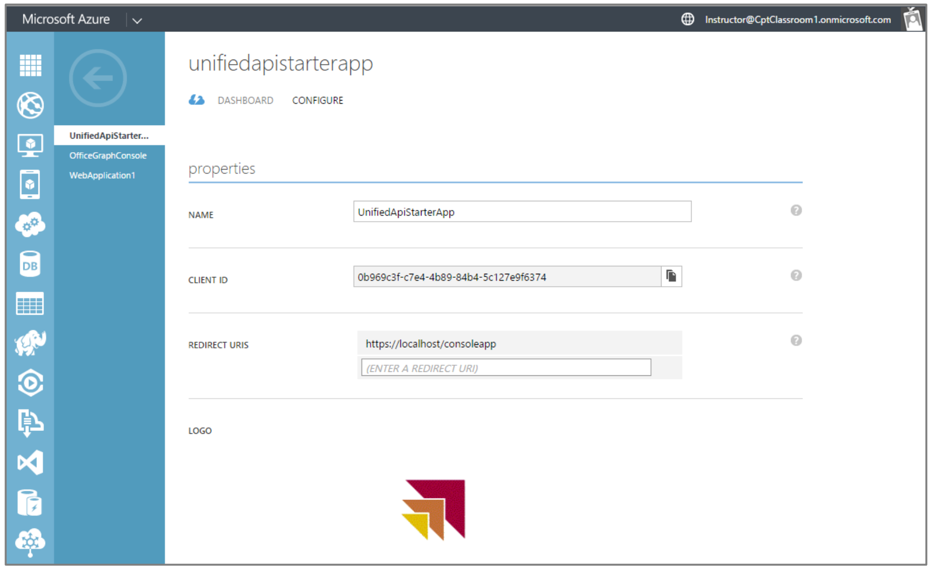
UnifiedApiStarterApp

Type

WEB APPLICATION AND/OR WEB API

[✓ NATIVE CLIENT APPLICATION]

Creating a new AAD Application


Microsoft Azure

Instructor@CptClassroom1.onmicrosoft.com

UnifiedApiStarter...

OfficeGraphConsole

WebApplication1

unifiedapistarterapp

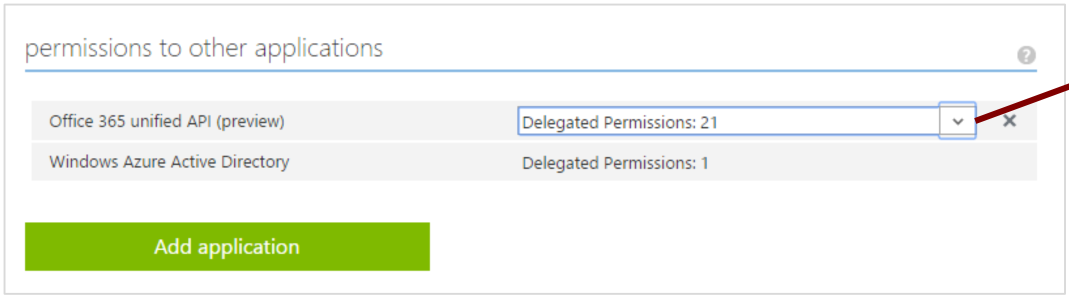
DASHBOARD CONFIGURE

properties

NAME	UnifiedApiStarterApp
CLIENT ID	0b969c3f-c7e4-4b89-84b4-5c127e9f6374
REDIRECT URIS	https://localhost/consoleapp (ENTER A REDIRECT URI)
LOGO	

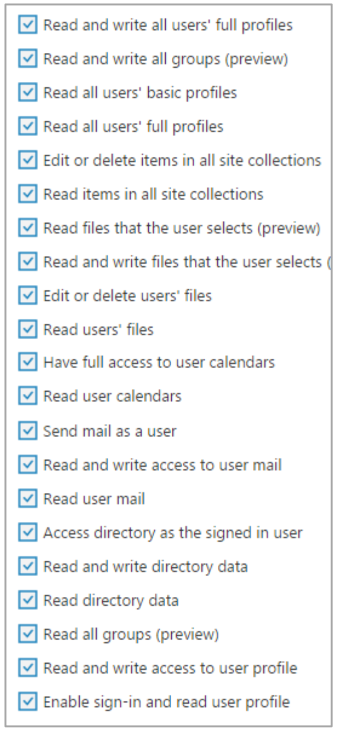
Configuring Unified API Permissions

- Application required permissions in AAD
  - Office 365 tools in Visual Studio tools not updated yet
  - Permissions configured in Azure Management Portal



permissions to other applications

Office 365 unified API (preview)	Delegated Permissions: 21
Windows Azure Active Directory	Delegated Permissions: 1



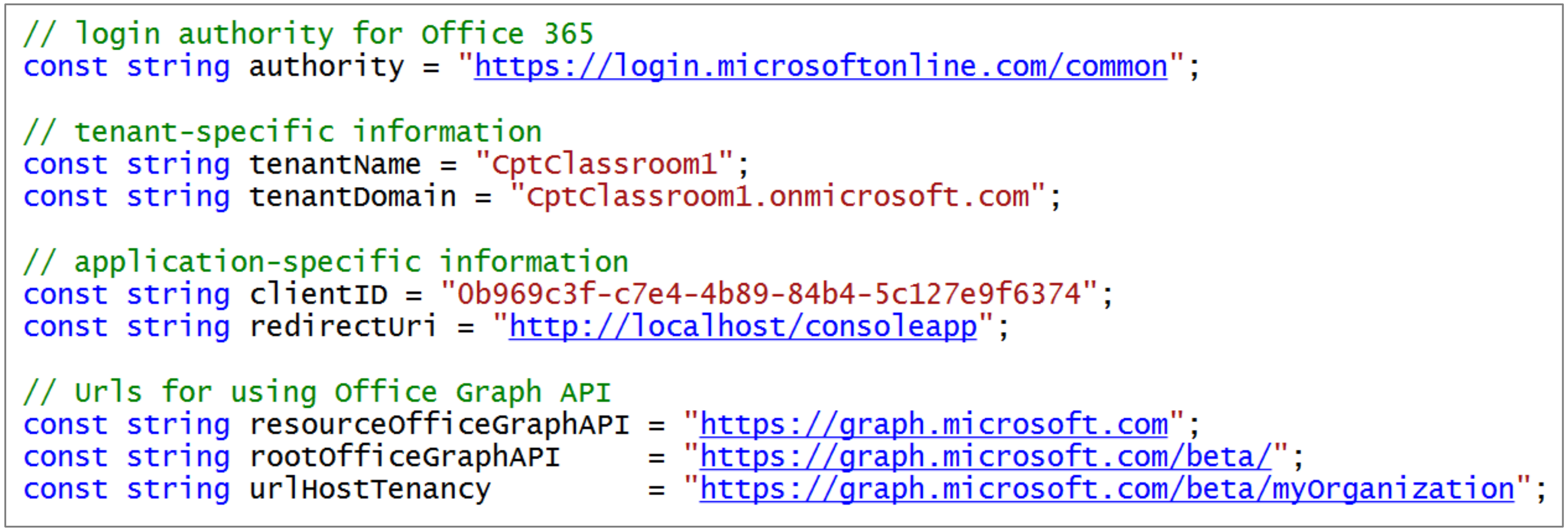
- ✓ Read and write all users' full profiles
- ✓ Read and write all groups (preview)
- ✓ Read all users' basic profiles
- ✓ Read all users' full profiles
- ✓ Edit or delete items in all site collections
- ✓ Read items in all site collections
- ✓ Read files that the user selects (preview)
- ✓ Read and write files that the user selects (
- ✓ Edit or delete users' files
- ✓ Read users' files
- ✓ Have full access to user calendars
- ✓ Read user calendars
- ✓ Send mail as a user
- ✓ Read and write access to user mail
- ✓ Read user mail
- ✓ Access directory as the signed in user
- ✓ Read and write directory data
- ✓ Read directory data
- ✓ Read all groups (preview)
- ✓ Read and write access to user profile
- ✓ Enable sign-in and read user profile

Agenda

- ✓ Introduction to the Office 365 APIs
- ✓ Office 365 API Service Endpoints
- ✓ Understanding the Microsoft Graph API
- ➢ Programming the Microsoft Graph API

Adding Application Constants

- Application requires tenant and app-specific information



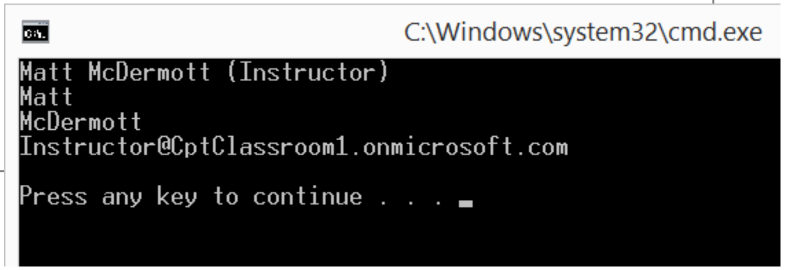
```
// login authority for Office 365
const string authority = "https://login.microsoftonline.com/common";

// tenant-specific information
const string tenantName = "CptClassroom1";
const string tenantDomain = "CptClassroom1.onmicrosoft.com";

// application-specific information
const string clientID = "0b969c3f-c7e4-4b89-84b4-5c127e9f6374";
const string redirectUri = "http://localhost/consoleapp";

// Urls for using Office Graph API
const string resourceOfficeGraphAPI = "https://graph.microsoft.com";
const string rootOfficeGraphAPI     = "https://graph.microsoft.com/beta/";
const string urlHostTenancy         = "https://graph.microsoft.com/beta/myOrganization";
```

Acquiring and Caching Access Tokens

```
class Program {

  "Demo constants"

  // add field to cache access token
  protected static string AccessToken = string.Empty;

  // add function to fetch and cache access token
  private static Task<string> AcquireTokenForUser() {

    // fetch access token from for Office Graph API if AccessToken is null
    if (string.IsNullOrEmpty(AccessToken)) {

      // create new authentication context
      var authenticationContext = new AuthenticationContext(authority, false);

      // use authentication context to trigger user sign-in and return access token
      var userAuthnResult = authenticationContext.AcquireToken(resourceOfficeGraphAPI,
                                                               clientID,
                                                               new Uri(redirectUri),
                                                               PromptBehavior.RefreshSession);

      // cache access token for reuse
      AccessToken = userAuthnResult.AccessToken;
    }
    // return access token to caller
    return Task.FromResult(AccessToken);
  }
```

Writing the "Hello World" Code

```
class Program {

  "Demo constants"

  // add field to cache access token
  protected static string AccessToken = string.Empty;

  // add function to fetch and cache access token
  private static Task<string> AcquireTokenForUser() ...

  static void Main() {

    //Create an GraphService object by passing in a service root and an access token.
    GraphService client = new GraphService(new Uri(urlHostTenancy), AcquireTokenForUser);

    // call across Internet and wait for response
    IUser user = client.Me.ExecuteAsync().Result;

    Console.WriteLine(user.displayName);
    Console.WriteLine(user.givenName);
    Console.WriteLine(user.surname);
    Console.WriteLine(user.mail);
    Console.WriteLine();
  }
}
```



```
Matt McDermott (Instructor)
Matt
McDermott
Instructor@CptClassroom1.onmicrosoft.com

Press any key to continue . . .
```

Office 365 Apps vs SharePoint Apps

- Points of comparison
  1. Types of applications
  2. Authentication Architecture
  3. Installation/deployment scope
  4. Permission granularity
  5. Launching an App
  6. Maturity of Platform
  7. Inline with Microsoft’s Strategy and direction

Learning Resources

- Microsoft online resources
  - http://Dev.Office.com
  - https://apisandbox.msdn.microsoft.com
- Office 365 Developer content on GitHub
  - https://github.com/OfficeDev
- On-demand Puralsight Course by Andrew Connell
  - Office 365 APIs: Overview, Authentication & the Discovery Service

Summary

- ✓ Introduction to the Office 365 APIs
- ✓ Office 365 API Service Endpoints
- ✓ Understanding the Microsoft Graph API
- ✓ Programming the Microsoft Graph API