Developing with the Power BI Service API

Setup Time: 60 minutes

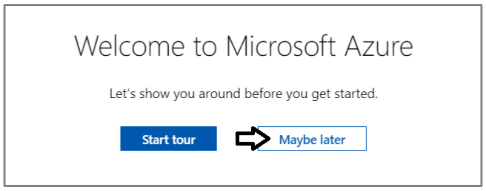
Lab Folder: C:\Student\Modules\02_PowerBiServiceAPI\Lab

Overview: In this lab, you will log into an Azure AD user account that has been created for you in an Azure AD tenant shared by all students which has a domain name of **powerbimvps.onmicrosoft.com**. Once you have logged into the Power BI service and started your 60 Power BI Pro trial, you will be able to upload PBIX files into your personal workspace and you can begin designing dashboard and reports in the browser. The lab will also step you through downloading and installing Power BI Desktop as well as publishing a Power BI Desktop project to the Power BI service. You create a new app workspace and populate it with a dataset, a report and a dashboard. In the final exercises, you will program against the Power BI Service API.

Exercise 1: Register a New Application with Azure Active Directory

In this exercise, you will register a new application with Azure AD and you will configure the application's required permissions to access the Power BI Service API.

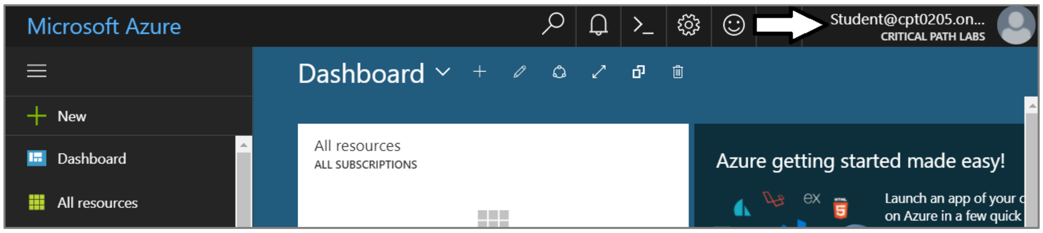
1. Log into the Azure Portal
   a) In the browser, navigate to the Azure portal at https://portal.azure.com.
   b) When you are prompted to log in, provide the credentials to log in with your Office 365 user account name.
   c) If you are prompted to start a tour of Microsoft Azure, click **Maybe later**.



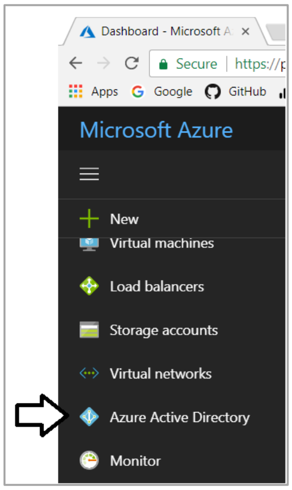
Welcome to Microsoft Azure

Let's show you around before you get started.

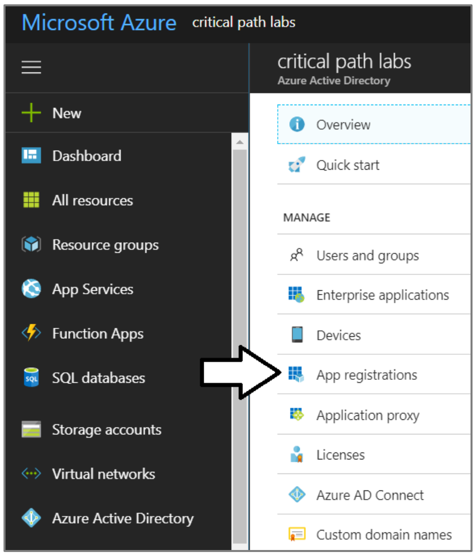
   d) Once you are log into the Azure portal, check the email address in the login menu in the upper right to make sure you are logged in the Azure portal with the correct identity.



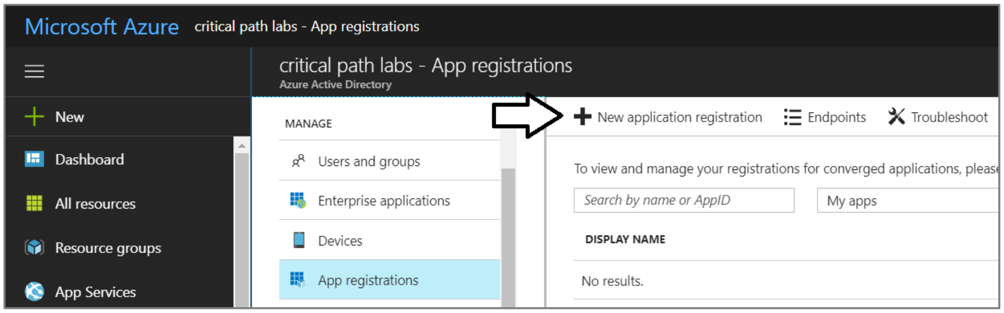
2. Register a new Azure application.
   a) In the left navigation, scroll down and click on the link for **Azure Active Directory**.



b) Click the link for **App registration**.



c) Click **New application registration**.



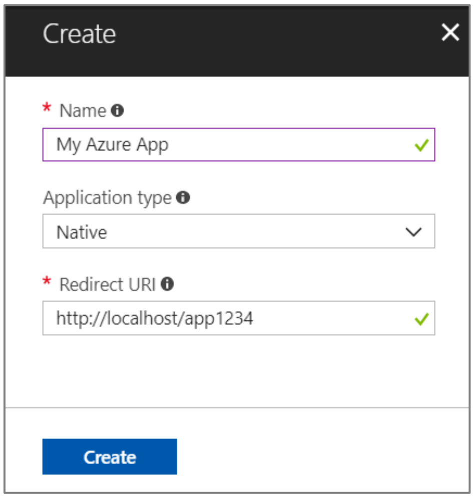
d) In the Create dialog…

i) Add a **Name** of **My Azure App**.

ii) Set the **Application type** to **Native**.

iii) Set the **Redirect URI** to http://localhost/app1234.

iv) Click the **Create** button to create the new application.

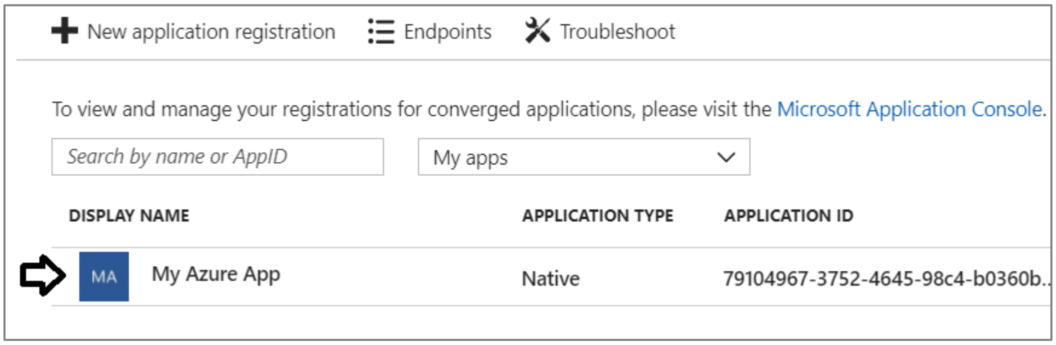


* Name: My Azure App

Application type: Native

* Redirect URI: http://localhost/app1234

e) Once should now see the new application.

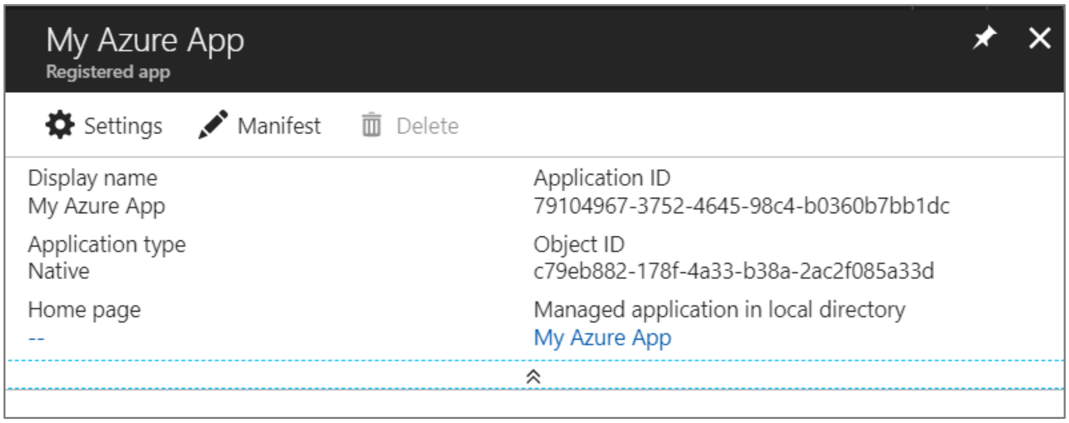


To view and manage your registrations for converged applications, please visit the Microsoft Application Console.

DISPLAY NAME	APPLICATION TYPE	APPLICATION ID
MA My Azure App	Native	79104967-3752-4645-98c4-b0360b..

3. Copy the GUID for the Application ID.

a) Click on the link for the new application named **My Azure App** to get to the details page.



Display name
My Azure App

Application ID
79104967-3752-4645-98c4-b0360b7bb1dc

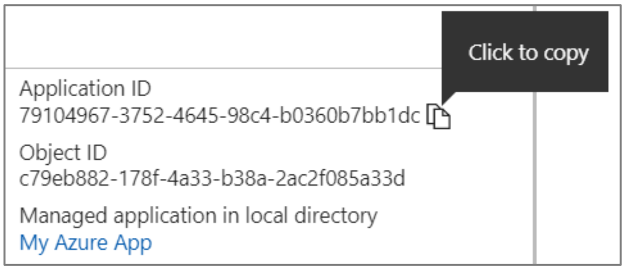
Application type
Native

Object ID
c79eb882-178f-4a33-b38a-2ac2f085a33d

Home page
--

Managed application in local directory
My Azure App

b) Copy the Application ID to the Windows clipboard.

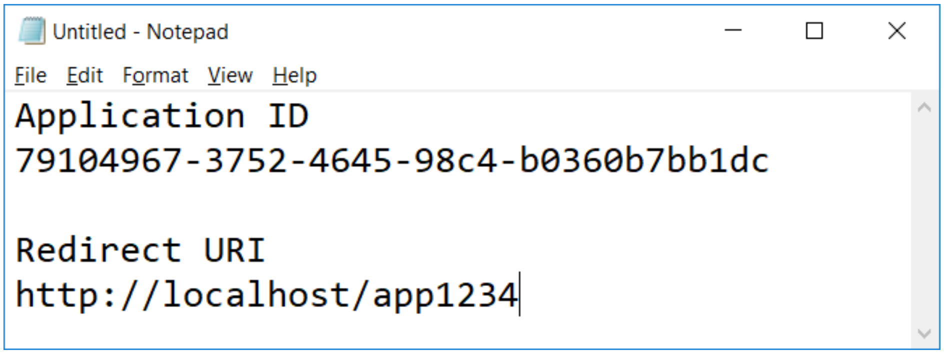


Application ID
79104967-3752-4645-98c4-b0360b7bb1dc

Object ID
c79eb882-178f-4a33-b38a-2ac2f085a33d

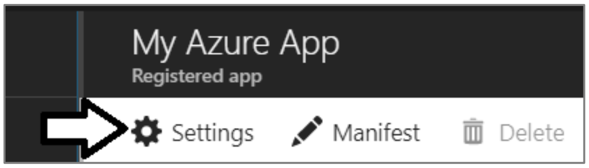
Managed application in local directory
My Azure App

c) Launch Notepad and paste the Application ID into a new document. Also add the value of the Redirect URI.

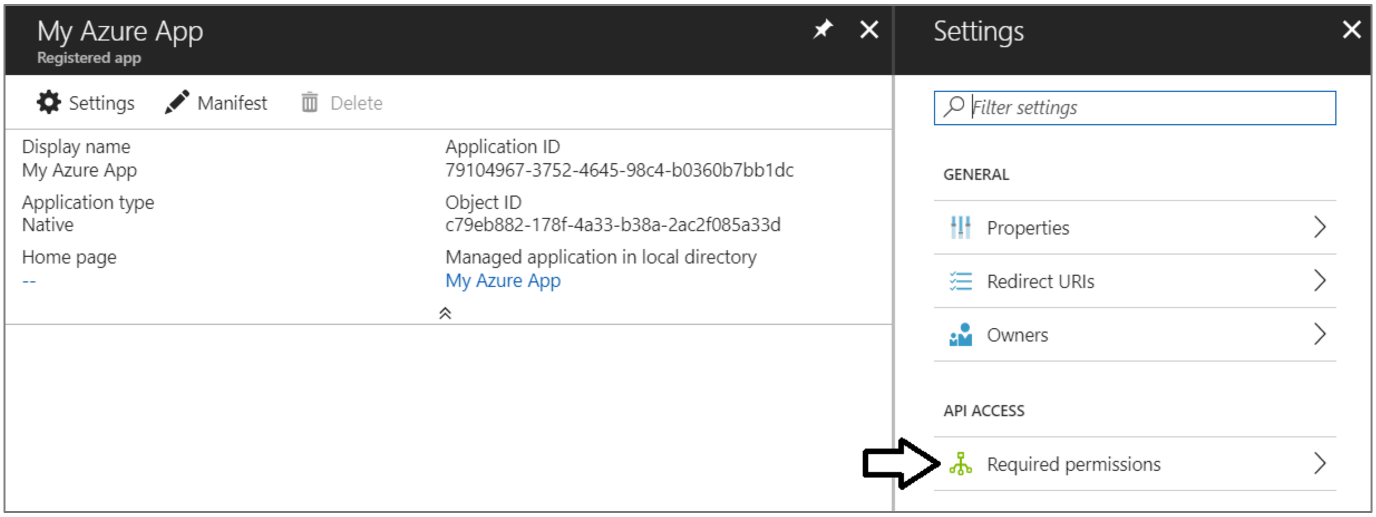


```
Application ID
79104967-3752-4645-98c4-b0360b7bb1dc

Redirect URI
http://localhost/app1234
```

d) Click on the **Settings** link to configure application settings,



e) Click **Required permissions**.



Display name
My Azure App

Application type
Native

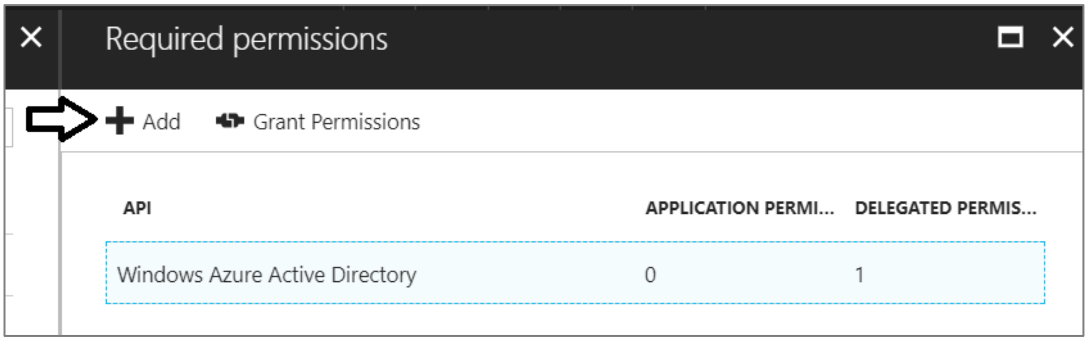
Home page
--

Application ID
79104967-3752-4645-98c4-b0360b7bb1dc

Object ID
c79eb882-178f-4a33-b38a-2ac2f085a33d

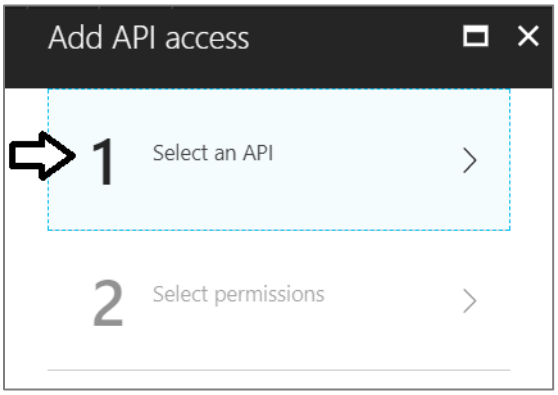
Managed application in local directory
My Azure App

f) Click the **Add** button on the **Required permissions** blade.

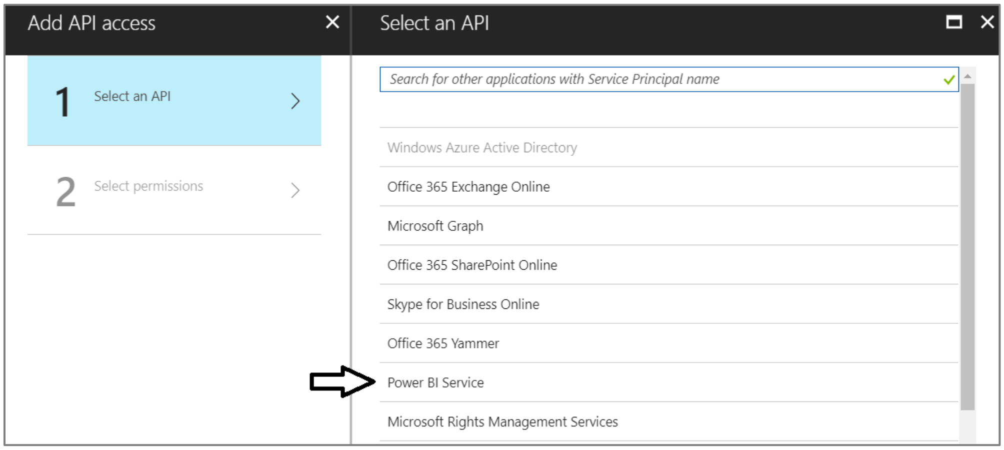


API	APPLICATION PERMI...	DELEGATED PERMIS...
Windows Azure Active Directory	0	1

g) Click the **Select an API** option in the **Add API** access blade.

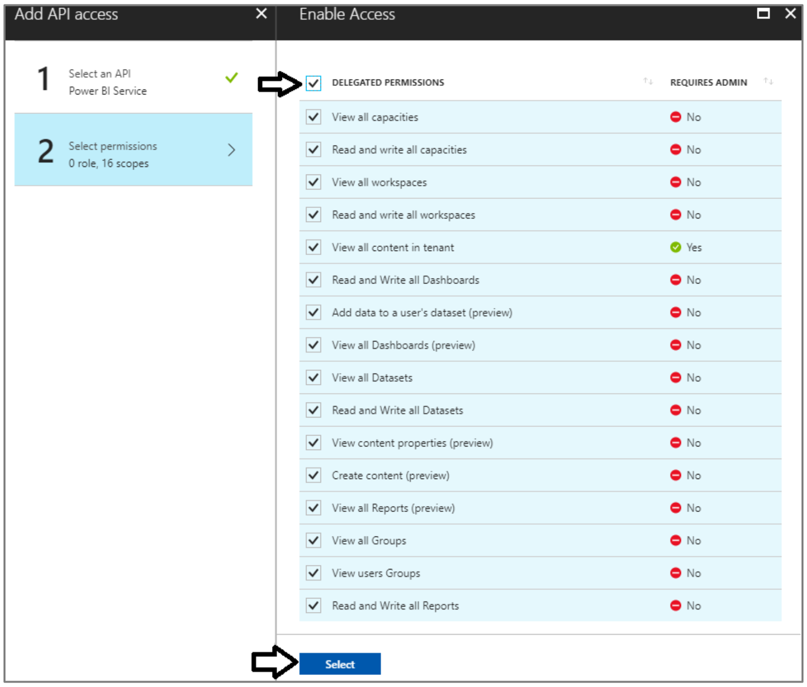


h) In the **Select an API** blade, click **Power BI Service**.


Add API access
1 Select an API
2 Select permissions
Select an API
Search for other applications with Service Principal name
Windows Azure Active Directory
Office 365 Exchange Online
Microsoft Graph
Office 365 SharePoint Online
Skype for Business Online
Office 365 Yammer
Power BI Service
Microsoft Rights Management Services

i) In the **Enable Access** blade, click the top checkbox for **DELEGATED PERMISSIONS** to select all the permissions.

j) Once you have selected all the permissions, click the **Select** button at the bottom of the blade.

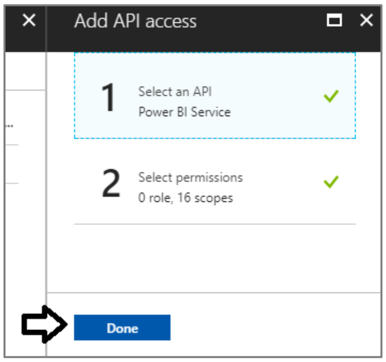


1 Select an API
Power BI Service ✓

2 Select permissions
0 role, 16 scopes

☑ DELEGATED PERMISSIONS	REQUIRES ADMIN
☑ View all capacities	⛔ No
☑ Read and write all capacities	⛔ No
☑ View all workspaces	⛔ No
☑ Read and write all workspaces	⛔ No
☑ View all content in tenant	✓ Yes
☑ Read and Write all Dashboards	⛔ No
☑ Add data to a user's dataset (preview)	⛔ No
☑ View all Dashboards (preview)	⛔ No
☑ View all Datasets	⛔ No
☑ Read and Write all Datasets	⛔ No
☑ View content properties (preview)	⛔ No
☑ Create content (preview)	⛔ No
☑ View all Reports (preview)	⛔ No
☑ View all Groups	⛔ No
☑ View users Groups	⛔ No
☑ Read and Write all Reports	⛔ No

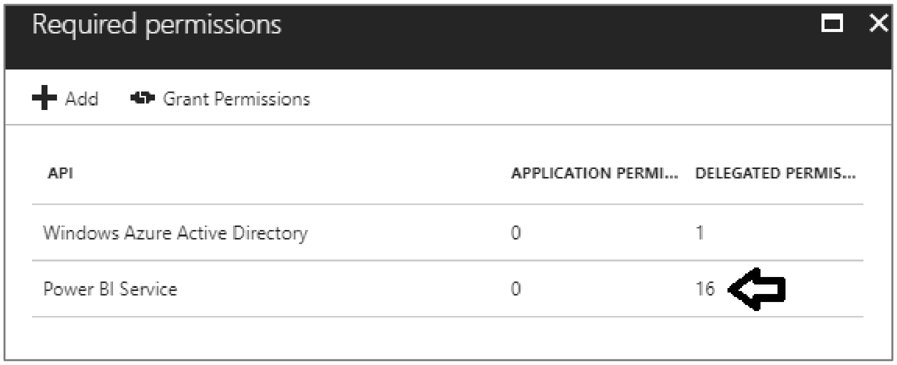
k) Click the **Done** button at the bottom of the **Add API Access** blade.



1 Select an API
Power BI Service ✓

2 Select permissions
0 role, 16 scopes ✓

l) At this point, you should be able to verify that the Power BI Service has been added to the **Required permissions** list.



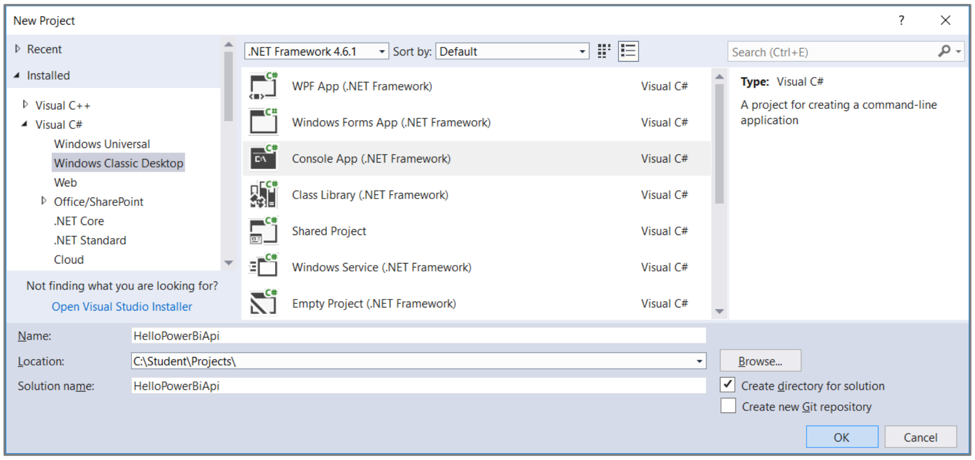
API	APPLICATION PERMI...	DELEGATED PERMIS...
Windows Azure Active Directory	0	1
Power BI Service	0	16

You are now done registering your application with Azure AD.

Exercise 2: Call the Power BI Service API using Direct REST Calls

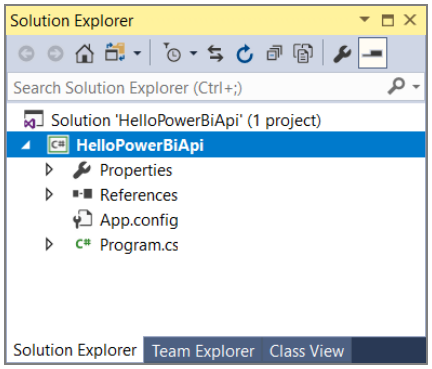
In this exercise, you will create a simple C# Console application to call into the Power BI Service API.

1. Create a new C# Console application in Visual Studio.
   a) Launch Visual Studio.
   b) Create a new project by running the **File > New Project** command.
   c) Select a project type of Console App from the Visual C# project templates.
   d) Give the project a name of **HelloPowerBiApi** and click OK.

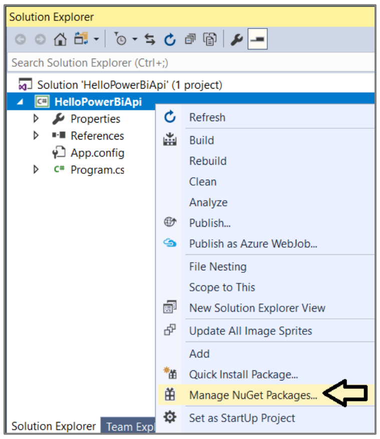

New Project
Recent
Installed
Visual C++
Visual C#
Windows Universal
Windows Classic Desktop
Web
Office/SharePoint
.NET Core
.NET Standard
Cloud
Not finding what you are looking for?
Open Visual Studio Installer
.NET Framework 4.6.1
Sort by: Default
Search (Ctrl+E)
WPF App (.NET Framework) Visual C#
Windows Forms App (.NET Framework) Visual C#
Console App (.NET Framework) Visual C#
Class Library (.NET Framework) Visual C#
Shared Project Visual C#
Windows Service (.NET Framework) Visual C#
Empty Project (.NET Framework) Visual C#
Type: Visual C#
A project for creating a command-line application
Browse...
OK
Cancel

Name: HelloPowerBiApi
Location: C:\Student\Projects\
Solution name: HelloPowerBiApi
☑ Create directory for solution
□ Create new Git repository

   e) You should now have a new project named **HelloPowerBiApi**.

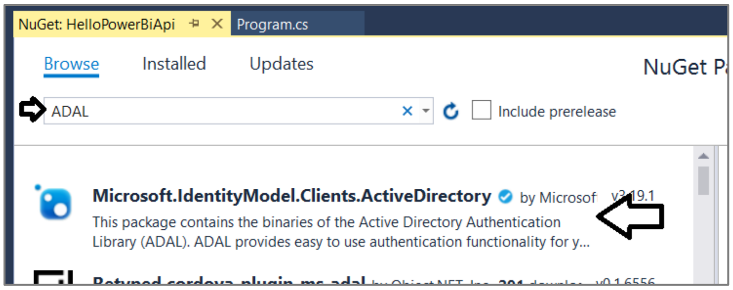


2. Add NuGet package to the project.
   a) Right-click the top-level node for the **HelloPowerBiApi** project and select **Manage NuGet Packages…**.



b) Click the Browse tab and type ADAL into the search box.

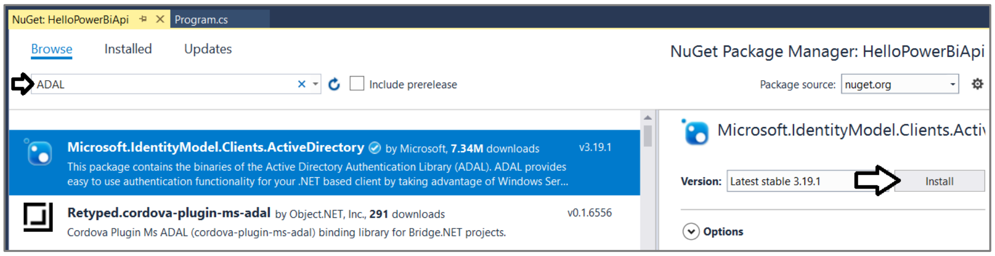
c) Locate the package **Microsoft.IdentityModel.Clients.ActiveDirectory**. This is the Active Directly Authentication library.



ADAL

Microsoft.IdentityModel.Clients.ActiveDirectory by Microsof v3.19.1
This package contains the binaries of the Active Directory Authentication Library (ADAL). ADAL provides easy to use authentication functionality for y...

d) Select and install **Microsoft.IdentityModel.Clients.ActiveDirectory**.



ADAL

Package source: nuget.org

Microsoft.IdentityModel.Clients.ActiveDirectory by Microsoft, 7.34M downloads v3.19.1
This package contains the binaries of the Active Directory Authentication Library (ADAL). ADAL provides easy to use authentication functionality for your .NET based client by taking advantage of Windows Ser...

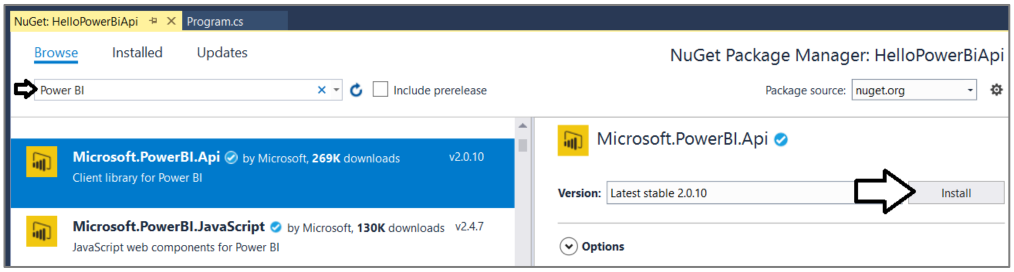
Retyped.cordova-plugin-ms-adal by Object.NET, Inc., 291 downloads v0.1.6556
Cordova Plugin Ms ADAL (cordova-plugin-ms-adal) binding library for Bridge.NET projects.

Microsoft.IdentityModel.Clients.Acti

Version: Latest stable 3.19.1

e) When prompted about the licensing agreement, click **I Agree**.

f) Search for Power BI and then find and install the **Microsoft.PowerBI.Api**.



Power BI

Package source: nuget.org

Microsoft.PowerBI.Api by Microsoft, 269K downloads v2.0.10
Client library for Power BI

Microsoft.PowerBI.JavaScript by Microsoft, 130K downloads v2.4.7
JavaScript web components for Power BI

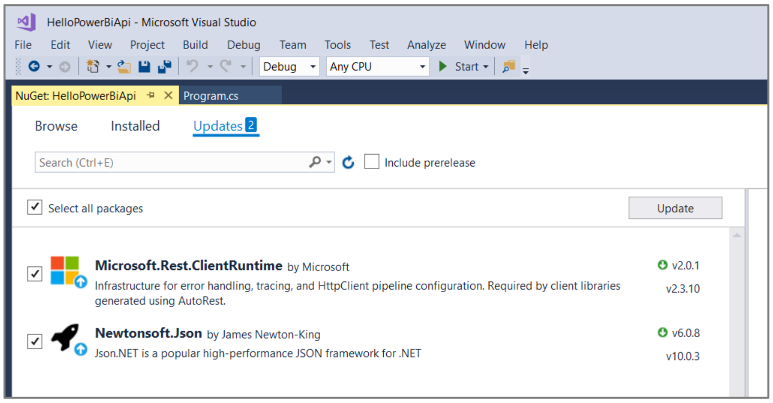
Microsoft.PowerBI.Api

Version: Latest stable 2.0.10

g) When prompted about the licensing agreement, click **I Agree**.

3. Update all NuGet packages.

a) Navigate to the **Update** tab and update any packages that have updates available.


HelloPowerBiApi - Microsoft Visual Studio
File Edit View Project Build Debug Team Tools Test Analyze Window Help
Debug
Any CPU
Start
NuGet: HelloPowerBiApi
Program.cs
Browse
Installed
Updates 2
Search (Ctrl+E)
Update

□ Include prerelease

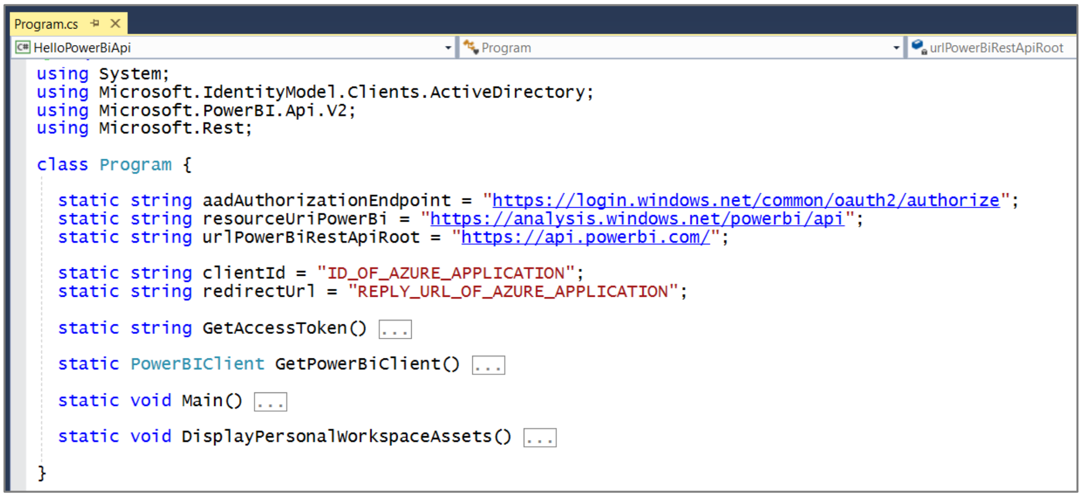
☑ Select all packages

☑ **Microsoft.Rest.ClientRuntime** by Microsoft — v2.0.1 / v2.3.10
Infrastructure for error handling, tracing, and HttpClient pipeline configuration. Required by client libraries generated using AutoRest.

☑ **Newtonsoft.Json** by James Newton-King — v6.0.8 / v10.0.3
Json.NET is a popular high-performance JSON framework for .NET

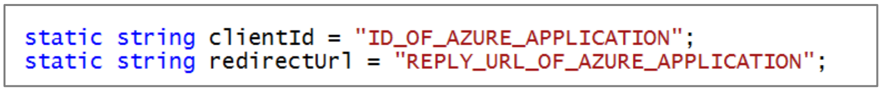
b) Close the window for the Nuget Package Manager.

4. Add code to the project
   a) Using Windows Explorer, locate the file named **PowerBiStarter.cs.txt** located in the Student folder at the following path.
   b) Open the file named **PowerBiStarter.cs.txt** in Notepad and copy its contents into the Window clipboard.
   c) Return to the **HelloPowerBiApi** project in Visual Studio.
   d) Open the source file named **program.cs**.
   e) Delete all the code inside **program.cs** and replace it with the content you copied into the Windows clipboard.
   f) You should now have the basic code for a simple application which access the Power BI Service API.



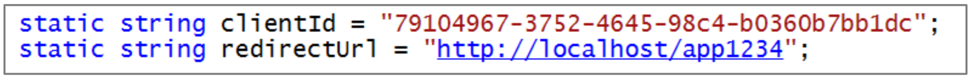
```
using System;
using Microsoft.IdentityModel.Clients.ActiveDirectory;
using Microsoft.PowerBI.Api.V2;
using Microsoft.Rest;

class Program {

  static string aadAuthorizationEndpoint = "https://login.windows.net/common/oauth2/authorize";
  static string resourceUriPowerBi = "https://analysis.windows.net/powerbi/api";
  static string urlPowerBiRestApiRoot = "https://api.powerbi.com/";

  static string clientId = "ID_OF_AZURE_APPLICATION";
  static string redirectUrl = "REPLY_URL_OF_AZURE_APPLICATION";

  static string GetAccessToken() ...

  static PowerBIClient GetPowerBiClient() ...

  static void Main() ...

  static void DisplayPersonalWorkspaceAssets() ...

}
```

5. Update the code with your Application ID and Redirect URI.
   a) Locate the section of the code with the static properties named **clientId** and **redirectUrl**.



```
static string clientId = "ID_OF_AZURE_APPLICATION";
static string redirectUrl = "REPLY_URL_OF_AZURE_APPLICATION";
```

   b) Replace these values with the values you copied into Notepad earlier.



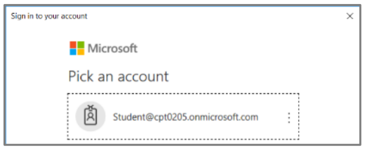
```
static string clientId = "79104967-3752-4645-98c4-b0360b7bb1dc";
static string redirectUrl = "http://localhost/app1234";
```

   c) Save your changes to **program.cs**.

6. Run the application to call to the Power BI Service API.

a) Press the {F5} key to begin a debugging session.

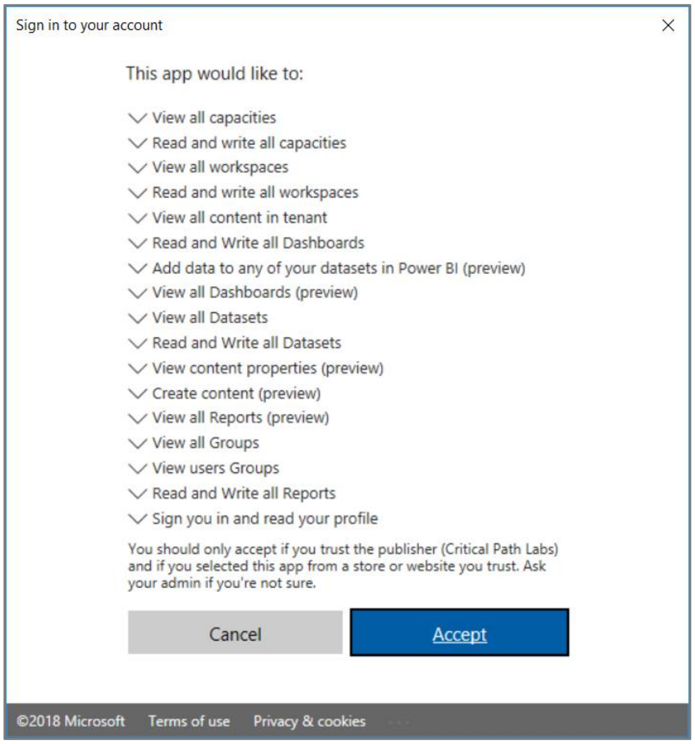
b) When promoted to sign in, log in using your Power BI account and credentials.



Pick an account

Student@cpt0205.onmicrosoft.com

c) When prompted with the following screen, click **Accept**.

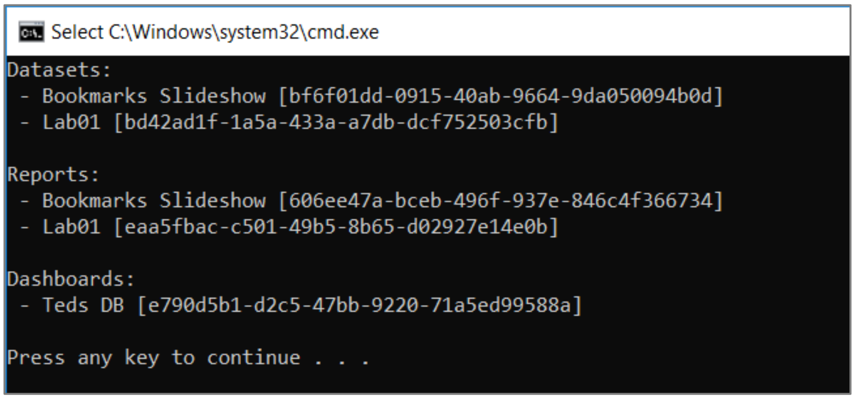


This app would like to:

- View all capacities
- Read and write all capacities
- View all workspaces
- Read and write all workspaces
- View all content in tenant
- Read and Write all Dashboards
- Add data to any of your datasets in Power BI (preview)
- View all Dashboards (preview)
- View all Datasets
- Read and Write all Datasets
- View content properties (preview)
- Create content (preview)
- View all Reports (preview)
- View all Groups
- View users Groups
- Read and Write all Reports
- Sign you in and read your profile

You should only accept if you trust the publisher (Critical Path Labs) and if you selected this app from a store or website you trust. Ask your admin if you're not sure.

d) The application should call into the Power BI Service API and retrieve data about the contents of your personal workspace.



```
Datasets:
 - Bookmarks Slideshow [bf6f01dd-0915-40ab-9664-9da050094b0d]
 - Lab01 [bd42ad1f-1a5a-433a-a7db-dcf752503cfb]

Reports:
 - Bookmarks Slideshow [606ee47a-bceb-496f-937e-846c4f366734]
 - Lab01 [eaa5fbac-c501-49b5-8b65-d02927e14e0b]

Dashboards:
 - Teds DB [e790d5b1-d2c5-47bb-9220-71a5ed99588a]

Press any key to continue . . .
```

Exercise 3: Call the Power BI Service API using the Power BI SDK

In this exercise, you will create a simple C# Console application to call into the Power BI Service API.

7. Create a new C# Console application in Visual Studio.

Exercise 4: Write C# Code to Upload a PBIX Project File

In this exercise, you will create a simple C# Console application to call into the Power BI Service API.

1. Create a new C# Console application in Visual Studio.

Exercise 5: Write C# Code to Copy Power BI Content Across Workspaces

In this exercise, you will create a simple C# Console application to call into the Power BI Service API.

1. Create a new C# Console application in Visual Studio.
2.

Congratulations. You have now successfully called into the Power BI Service API.