



Microsoft Power BI Blog


BLOG > ANNOUNCEMENTS > DEVELOPERS > FEATURES

Working with PowerShell in Power BI



Kay Unkroth

Senior Program Manager

 August 13, 2018



If you are following the [Power BI blog](#) on a regular basis, you probably have noticed the [Power BI APIs and cmdlets announcement for administrators](#), which introduced a set of APIs and cmdlets to work with workspaces, dashboards, reports, datasets, and so forth in Power BI. But there is much more to this than could be covered in a brief announcement. For starters, the management cmdlets are not just for administrators; they are also for Power BI users and developers. This article takes a closer look to show you how to take advantage of these cmdlets provided your profile fits any of the following target groups:

- **Power BI administrators**, who must work with resources at the Power BI tenant level, such as to enumerate workspaces, dashboards, reports, and datasets, recover orphaned workspaces, or discover the data sources used in the various datasets across an entire organization.
- **Power BI users**, who want to manage the resources they own in Power BI, such as to create a backup of all published reports across multiple workspaces.
- **Power BI developers**, who need to have a quick and easy way to prototype their solutions, such as by trying out custom REST API calls in the PowerShell console without having to resort to Web-based prototyping tools and copying access tokens.
- **Power BI community members**, who wish to contribute to the benefit of others, such as

Power BI



Let's cover these points in reverse order beginning with the community aspect. Power BI enjoys a vibrant community committed to helping each other getting the most out of Power BI. Community members frequently trailblaze through uncharted territory with solutions and tools that complement what's available from Microsoft. PowerShell support for Power BI is a good example. Search for "*power bi powershell*" and you can find plenty of articles showing you how to use the Power BI APIs from the PowerShell console. You can even find a `Microsoft.PowerBI.PowerShell` module in the PowerShell Gallery and a collection of cmdlets for working with the PowerBI APIs on GitHub (<https://github.com/DevScope/powerbi-powershell-modules>). This is great, yet it creates a slight dilemma because cmdlets from different sources don't work well together and they duplicate functionality while leaving gaps in other areas. It is much better to join forces, coordinate the work, merge what already exists, and deliver the richest possible collection of cmdlets consolidated in a single repository.

You can find this Microsoft-backed GitHub repo at <https://github.com/Microsoft/powerbi-powershell>. If you are interested in helping build cmdlets for Power BI, check out [Contribute Code to PowerShell Cmdlets for Power BI](#) and come on board as a project member. Or simply report any issues you encounter at <https://github.com/Microsoft/powerbi-powershell/issues>. All help is greatly appreciated.

One advantage of the Power BI management cmdlets is that they are available as a [module from the PowerShell Gallery](#). Simply install them by using the command **Install-Module -Name MicrosoftPowerBIMgmt -Scope CurrentUser**. Another is that the management cmdlets do not require a separate app registration in Azure. So, if you are a Power BI app developer, it's very straightforward and quick to prototype solutions that make Power BI Rest API calls. Just log in to Power BI by using the **Login-PowerBI** cmdlet. It supports user accounts, service principals, and application credentials (key or certificate). Then fire away by using one of the pre-built cmdlets or use the **Invoke-PowerBIRestMethod** cmdlet.

Invoke-PowerBIRestMethod is the Swiss army knife of Power BI cmdlets. You can use it to construct arbitrary REST API calls against Power BI. So, if you find that you are missing a specific management cmdlet, you can still get the job done if there is an appropriate REST API endpoint for you to call. For all the details about available REST API endpoints, see the [Power BI REST API reference](#).

For working with workspaces, dashboards, reports, datasets, data sources, and imports, you don't need to go to the **Invoke-PowerBIRestMethod** cmdlet. Instead, use the more specialized cmdlets **Get-PowerBIWorkspace**, **Get-PowerBIReport**, **Get-PowerBIDataset**, and so forth. These cmdlets are more convenient. Also, take a moment to check out the online help, which you can access through the **Get-Help** cmdlet. For example, use the command **Get-Help Get-**

Power BI



parameter called **-Scope**. This parameter can take one of two possible values: **Individual** and **Organization**. The default is **Individual**, which only returns those items that you can access as a regular Power BI user. On the other hand, if you are working as a Power BI admin, specify **Organization** to return all the items of a given type that exist in your Power BI tenant. Only Power BI users with admin rights (such as Office 365 Global Administrator or Power BI Service Administrator) can use the organization scope, but all users can use the individual scope, so the management cmdlets can really help any Power BI user automate repetitive tasks.

The following PowerShell script illustrates the difference between the individual and organization scope. The script illustrates both the use of the **Get-PowerBIWorkspace** cmdlet as well as comparable requests based on the **Invoke-PowerBIRestMethod** cmdlet, highlighting the fact that the individual scope invokes the user API whereas the organization scope relies on the admin API.

See also the screenshot below the listing for actual results in a test tenant.

```
## Login to Power BI using an account with service admin rights.
```

```
##
```

```
Login-PowerBI
```

```
## Get the list of workspaces as a Power BI user.
```

```
##
```

```
$myWorkspaces = Get-PowerBIWorkspace
```

```
Write-Host "The current user has --" $myWorkspaces.Count "-- workspaces."
```

```
## Get the list of workspaces as a Power BI admin.
```

```
##
```

```
$tenantWorkspaces = Get-PowerBIWorkspace -Scope Organization
```

```
Write-Host "There are --" $tenantWorkspaces.Count "-- workspaces in this Power BI tenant."
```

```
## Get the list of workspaces by making a direct REST API call against the user API.
```

```
## /v1.0/myorg/groups
```

```
##
```

```
$myRestResult = Invoke-PowerBIRestMethod -Url 'Groups' -Method Get | ConvertFrom-Json
```

```
Write-Host "The current user has --" $myRestResult.value.Count "-- workspaces."
```

Power BI

##

```
$tenantRestResult = Invoke-PowerBIRestMethod -Url 'admin/Groups' -Method Get |
ConvertFrom-Json
```

```
Write-Host "There are --" $tenantRestResult.value.Count "-- workspaces in this Power BI tenant."
```

```
Windows PowerShell ISE
File Edit View Tools Debug Add-ons Help
IndividualvsOrganization.ps1 X
1 ## Login to Power BI using an account with service admin rights.
2 ##
3 Login-PowerBI
4
5 ## Get the list of workspaces as a Power BI user.
6 ##
7 $myWorkspaces = Get-PowerBIWorkspace
8 Write-Host "The current user has --" $myWorkspaces.Count "-- workspaces."
9
10 ## Get the list of workspaces as a Power BI admin.
11 ##
12 $tenantWorkspaces = Get-PowerBIWorkspace -Scope Organization
13 Write-Host "There are --" $tenantWorkspaces.Count "-- workspaces in this Power BI tenant."
14
15 ## Get the list of workspaces by making a direct REST API call against the user API.
16 ## /v1.0/myorg/groups
17 ##
18 $myRestResult = Invoke-PowerBIRestMethod -Url 'Groups' -Method Get | ConvertFrom-Json
19 Write-Host "The current user has --" $myRestResult.value.Count "-- workspaces."
20
21 ## Get the list of workspaces by making a direct REST API call against the admin API.
22 ## /v1.0/myorg/admin/groups
23 ##
24 $tenantRestResult = Invoke-PowerBIRestMethod -Url 'admin/Groups' -Method Get | ConvertFrom-Json
25 Write-Host "There are --" $tenantRestResult.value.Count "-- workspaces in this Power BI tenant."

PS C:\Users\kayu\Desktop> C:\Users\kayu\Desktop\IndividualvsOrganization.ps1

Environment : Public
TenantId    : A9B3366C-308A-4EBB-AC8C-A6A88357AC25
UserName    : TestAdmin@testtenant.onmicrosoft.com

The current user has -- 25 -- workspaces.
There are -- 416 -- workspaces in this Power BI tenant.
The current user has -- 25 -- workspaces.
There are -- 416 -- workspaces in this Power BI tenant.

PS C:\Users\kayu\Desktop>
```

What happens if you call an admin API as a regular user without admin rights? As expected, PowerShell will display an error message informing you that the call was unauthorized. If you are among those users who would like to dig a little deeper, use the **Resolve-PowerBIErrors** cmdlet. If you run this cmdlet without any parameters, it shows the details for all errors that might have occurred in your current PowerShell session. You can specify **-Last** as a parameter to analyze only the last error or reference the desired error directly in the **-Error** parameter, such as **-Error**

Power BI

```
## Login as a regular Power BI user without service admin rights.
```

```
##
```

```
Login-PowerBI
```

```
## Try to work with the organization scope.
```

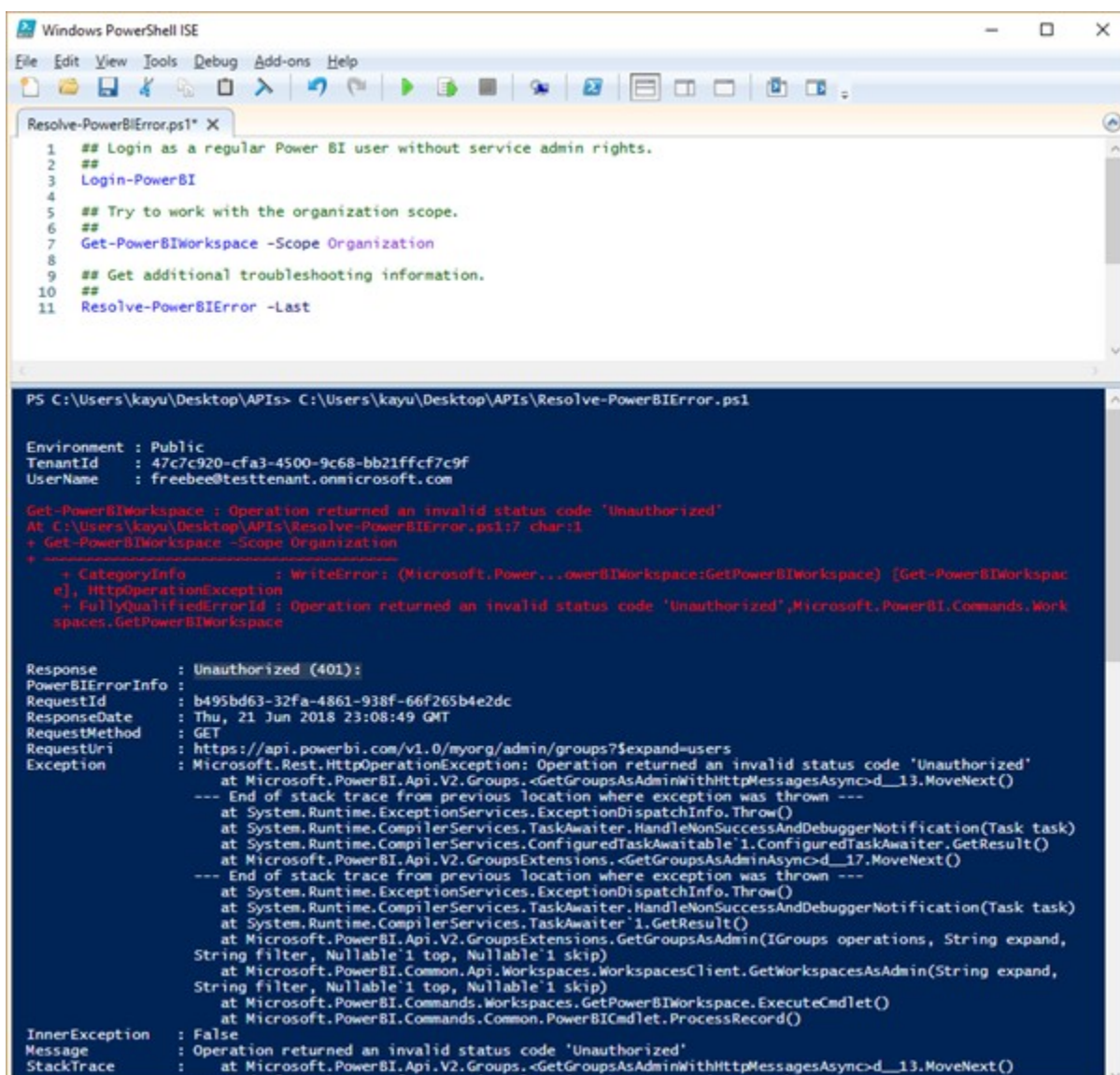
```
##
```

```
Get-PowerBIWorkspace -Scope Organization
```

```
## Get additional troubleshooting information.
```

```
##
```

```
Resolve-PowerBIErrors -Last
```



The screenshot shows a Windows PowerShell ISE window with a script named 'Resolve-PowerBIErrors.ps1' loaded. The script contains 11 lines of PowerShell commands. The output pane shows the execution of the script, starting with the environment information and then the execution of the 'Get-PowerBIWorkspace -Scope Organization' command. This command results in an 'Unauthorized (401)' error. The output pane displays the error details, including the request ID, response date, request method, and request URI. The exception message is 'Operation returned an invalid status code 'Unauthorized''. The stack trace shows the call path from the 'Resolve-PowerBIErrors' command through the 'Get-PowerBIWorkspace' command to the 'GetGroupsAsAdmin' method.

```
Windows PowerShell ISE
File Edit View Tools Debug Add-ons Help

Resolve-PowerBIErrors.ps1 X
1 ## Login as a regular Power BI user without service admin rights.
2 ##
3 Login-PowerBI
4
5 ## Try to work with the organization scope.
6 ##
7 Get-PowerBIWorkspace -Scope Organization
8
9 ## Get additional troubleshooting information.
10 ##
11 Resolve-PowerBIErrors -Last

PS C:\Users\kayu\Desktop\APIs> C:\Users\kayu\Desktop\APIs\Resolve-PowerBIErrors.ps1

Environment : Public
TenantId    : 47c7c920-cfa3-4500-9c68-bb21ffc7c9f
UserName    : freebee@testtenant.onmicrosoft.com

Get-PowerBIWorkspace : Operation returned an invalid status code 'Unauthorized'
At C:\Users\kayu\Desktop\APIs\Resolve-PowerBIErrors.ps1:7 char:1
+ Get-PowerBIWorkspace -Scope Organization
+ ~~~~~
+ CategoryInfo          : WriteError: (Microsoft.PowerShell.PowerBIWorkspace:GetPowerBIWorkspace) [Get-PowerBIWorkspace]
+ FullyQualifiedErrorId : Operation returned an invalid status code 'Unauthorized',Microsoft.PowerShell.Commands.Workspaces.GetPowerBIWorkspace

Response           : Unauthorized (401):
PowerBIErrorsInfo :
RequestId          : b495bd63-32fa-4861-938f-66f265b4e2dc
ResponseDate       : Thu, 21 Jun 2018 23:08:49 GMT
RequestMethod       : GET
RequestUri         : https://api.powerbi.com/v1.0/myorg/admin/groups?expand=users
Exception          : Microsoft.Rest.HttpOperationException: Operation returned an invalid status code 'Unauthorized'
                    at Microsoft.PowerBI.Api.V2.Groups.<GetGroupsAsAdminWithHttpMessagesAsync>d__13.MoveNext()
                    --- End of stack trace from previous location where exception was thrown ---
                    at System.Runtime.ExceptionServices.ExceptionDispatchInfo.Throw()
                    at System.Runtime.CompilerServices.TaskAwaiter.HandleNonSuccessAndDebuggerNotification(Task task)
                    at System.Runtime.CompilerServices.TaskAwaiter`1.ConfiguredTaskAwaitable`1.GetResult()
                    at Microsoft.PowerBI.Api.V2.GroupsExtensions.<GetGroupsAsAdminAsync>d__17.MoveNext()
                    --- End of stack trace from previous location where exception was thrown ---
                    at System.Runtime.ExceptionServices.ExceptionDispatchInfo.Throw()
                    at System.Runtime.CompilerServices.TaskAwaiter.HandleNonSuccessAndDebuggerNotification(Task task)
                    at System.Runtime.CompilerServices.TaskAwaiter`1.GetResult()
                    at Microsoft.PowerBI.Api.V2.GroupsExtensions.GetGroupsAsAdmin(IGroups operations, String expand,
String filter, Nullable`1 top, Nullable`1 skip)
                    at Microsoft.PowerBI.Common.Api.Workspaces.WorkspacesClient.GetWorkspacesAsAdmin(String expand,
String filter, Nullable`1 top, Nullable`1 skip)
                    at Microsoft.PowerBI.Commands.Workspaces.GetPowerBIWorkspace.ExecuteCmdlet()
                    at Microsoft.PowerBI.Commands.Common.PowerBICmdlet.ProcessRecord()

InnerException      : False
Message             : Operation returned an invalid status code 'Unauthorized'
StackTrace           : at Microsoft.PowerBI.Api.V2.Groups.<GetGroupsAsAdminWithHttpMessagesAsync>d__13.MoveNext()
```


Power BI



with a brief list of typical tasks that the management cmdlets can help to accomplish efficiently:

- Automating recurring tasks using arbitrary REST API calls and third-party cmdlets without having to register an application in Azure—Log in to Power BI (**Login-PowerBI**), send arbitrary REST API calls to Power BI (**Invoke-PowerBIRestMethod**), retrieve the access token and pass it to any third-party cmdlets that accept an access token (**Get-PowerBIAccessToken**), and eventually log out (**Logout-PowerBI**).
- Enumerating all workspaces, dashboards, tiles, reports, datasets, data sources, and imports in a Power BI tenant—Log in to Power BI as an admin, and then use the cmdlets **Get-PowerBIWorkspace**, **Get-PowerBIDashboard**, **Get-PowerBITile**, **Get-PowerBIReport**, **Get-PowerBIDataset**, **Get-PowerBIDatasource**, and **Get-PowerBIImport**. Do not forget to specify **-Scope Organization**.
- Exporting all reports in a Power BI tenant to local files—Log in to Power BI as an admin, and then use the **Export-PowerBIReport** cmdlet. Do not forget to specify **-Scope Organization** and make sure you export each report into a separate file.
- Restore a deleted workspace—Login to Power BI, and then use the **Get-PowerBIWorkspace** cmdlet to enumerate the deleted workspaces and the **Restore-PowerBIWorkspace** cmdlet to restore them. Be aware that restoring workspaces is currently limited to the new workspace experiences preview.
- Renaming a workspace and adding or removing users—Log in to Power BI, and then use the **Get-PowerBIWorkspace**, **Set-PowerBIWorkspace**, **Add-PowerBIWorkspaceUser**, and **Remove-PowerBIWorkspaceUser** cmdlets. If you are updating a workspace you own, you don't need to specify the organization scope, but be aware that updating workspaces is currently limited to the new workspace experiences preview.
- Enumerating workspaces that have a given user as a member—Log in to Power BI as an admin, and then use the **Get-PowerBIGroup** cmdlet with the **-User** parameter. Do not forget to specify **-Scope Organization** and be aware that membership information is currently limited to the new workspace experiences preview, so the command will not return personal or group-based legacy workspaces.
- Enumerating all orphaned workspaces (i.e. workspaces without an owner) in a Power BI tenant—Log in to Power BI as an admin, and then use the **Get-PowerBIGroup** cmdlet with the **-Orphaned** parameter. Do not forget to specify **-Scope Organization**. Again, this capability is currently limited to the new workspace experiences preview, so the command will not return any orphaned personal or group-based legacy workspaces.

Power BI



based on the capacity ID, such as **Get-PowerBIGroup -Filter "capacityId eq '05801684-baba-4de0-8a6b-5445cf7df011'" -Scope Organization**. Then pipe the output to **Get-PowerBIDashboard**, **Get-PowerBITile**, **Get-PowerBIReport**, **Get-PowerBIDataset**, **Get-PowerBIDatasource**, and **Get-PowerBIImport** to retrieve only the relevant resources from these workspaces. Do not forget to specify **-Scope Organization** because the **-Filter** parameter is currently not supported in the individual scope.

That's it for the moment. As mentioned, subsequent blog posts will dive deeper into some of these scenarios. So, stay tuned for more information about how to work with Power BI in the PowerShell console. And, as always, please send us your feedback via any of the available communication channels such as UserVoice and community forums or simply go to <https://github.com/Microsoft/powerbi-powershell/issues>. You can influence the evolution of the Power BI management cmdlets to the benefit of all Power BI users!

APIS

ENUMERATING DATASETS AND DATA SOURCES

MANAGEMENT CMDLETS

POWER BI ADMIN

SIGN UP FOR THE POWER BI NEWSLETTER

Sign up below to get the latest from Power BI, direct to your inbox!

Country

By clicking Sign up today, you are giving your consent to Microsoft for the Power BI newsletter program to provide you the exclusive news, surveys, tips and advice and other information for getting the most out of Power BI. You can unsubscribe at any time. [Microsoft Privacy Statement](#).

SUBMIT

Power BI



ALSO ON THE POWER BI BLOG

Join us! Dataflows Webinar Mar 12 ...

a month ago • 3 comments

<p>Create custom visuals using R, JSON scripts and Charticulator. </p>

Announcing read/write XMLA endpoints in ...

a month ago • 35 comments

<p>We are excited to announce the public preview of read/write ...

On-premises data gateway March 2020 ...

a month ago • 5 comments

<p>March Gateway release</p>

Join us 11:00 a

22 days a

<p>Powe
flowing th
a Busine

Comments for this thread are now closed



4 Comments

The Power BI blog

Disqus' Privacy Policy

Login ▾

Recommend 2

Tweet

Share

Sort by Oldest ▾



Tate • 2 years ago • edited

This is a great start, and is going to be massively useful. We'd love to see some built-in cmdlets for Gateway administration. It doesn't look like there are any on the roadmap, but perhaps this is something still being discussed.

<https://github.com/Microsoft...>

For now, we'll resort to the Swiss army knife. :)

^ | ▾ • Share ›



Leonardo Araujo • 2 years ago

very nice, thanks

^ | ▾ • Share ›



Hongju Jung • 2 years ago

I execute Export-PowerBIReport as Power Admin
(Export-PowerBIReport -Id "1d70184d-92df-4d41-bfce-f087d244f42e" -OutFile "test.pbix")
I got error but I can get a file on Power Service.
(Export-PowerBIReport : Operation returned an invalid status code 'Unauthorized')

^ | ▾ • Share ›



Chris • 2 years ago

Hello, when is a Power BI Pro license required for Power BI REST API access? For example, is it required to access the new admin methods (such as GetGroupsAsAdmin)? Thanks

Power BI



WHAT IS POWER BI?

Power BI is a suite of business analytics tools to analyze data and share insights. Monitor your business and get answers quickly with rich dashboards available on every device.

[READ MORE](#)

SUBSCRIBE TO THE POWER BI BLOG

 [SUBSCRIBE](#)

SEARCH BY CATEGORY

Analysis Services

Announcements

API

Developers

Features

Paginated Reports

Power BI

Power BI Embedded

Reporting Services

Support

Template apps

Power BI



March 2020

February 2020

January 2020

December 2019

November 2019

October 2019

September 2019

August 2019

July 2019

June 2019

May 2019

April 2019

March 2019

February 2019

January 2019

December 2018

November 2018

October 2018

September 2018

August 2018

July 2018

June 2018

May 2018

April 2018

March 2018

February 2018

Power BI



November 2017

October 2017

September 2017

August 2017

July 2017

June 2017

May 2017

April 2017

March 2017

February 2017

January 2017

December 2016

November 2016

October 2016

September 2016

August 2016

July 2016

June 2016

May 2016

April 2016

March 2016

February 2016

January 2016

December 2015

November 2015

October 2015

Power BI



July 2015

June 2015

May 2015

April 2015

March 2015

February 2015

January 2015

December 2014

November 2014

October 2014

September 2014

August 2014

July 2014

June 2014

May 2014

April 2014

March 2014

February 2014

January 2014

December 2013

November 2013

October 2013

September 2013

August 2013

July 2013

Power BI



POWER PLATFORM

[Overview](#)

[Power BI](#)

[Power Apps](#)

[Power Automate](#)

[Power Virtual Agents](#)

[Sign in](#)

[Sign up](#)

BROWSE

[Solutions](#)

[Partners](#)

[Consulting Services](#)

DOWNLOADS

[Power BI Desktop](#)

[Power BI Mobile](#)

[Power BI Report Server](#)

[See all downloads](#)

LEARN

[Guided learning](#)

[Documentation](#)

[Support](#)

[Community](#)

Power BI



Developers

Blog

Newsletter



© 2020 Microsoft



[Privacy & cookies](#)

[Terms of use](#)

[Trademarks](#)

[English \(US\)](#)



[Request demo](#)