## Writing PowerShell Scripts for Power BI

In this lab, you will create a.

To complete this lab, your developer workstation must be configured to allow the execution of PowerShell scripts. Your developer workstation must also have the following software and developer tools installed.

1) **PowerShell cmdlet library for AzureAD** – [[download](https://docs.microsoft.com/en-us/powershell/azure/active-directory/install-adv2?view=azureadps-2.0)]

2) **DOTNET Core SDK 3.1 or later** – [[download](https://dotnet.microsoft.com/download)]

3) **Node.js** – [[download](https://nodejs.org/en/download/)]

4) **Visual Studio Code** – [[download](https://code.visualstudio.com/Download)]

5) **Visual Studio 2019** (optional) – [[download](https://visualstudio.microsoft.com/downloads/)]

.

### Exercise 1: Configure PowerShell to Run Scripts on Your Computer

In this exercise, xxx.

1. Download the student lab files to a local folder on your developer workstation.
   1. Create a new top-level folder on your workstation named **DevCamp** at a location such as **c:\DevCamp**.
   2. Download the ZIP archive with the student lab files from GitHub by clicking the following link.

<https://github.com/PowerBiDevCamp/PowerBI-PowerShell-Tutorial/archive/master.zip>

* 1. Extract the **StudentLabFiles** folder from **StudentLabFiles.zip** into a to a local folder such as **c:\DevCamp\StudentLabFiles**.
  2. The **StudentLabFiles** folder should contain the set of files shown in the following screenshot.

Clear-Host

$hobbies = @("Pilates", "Kick boxing", "Power BI Embedding")

Write-Host

Write-Host "My Hobbies"

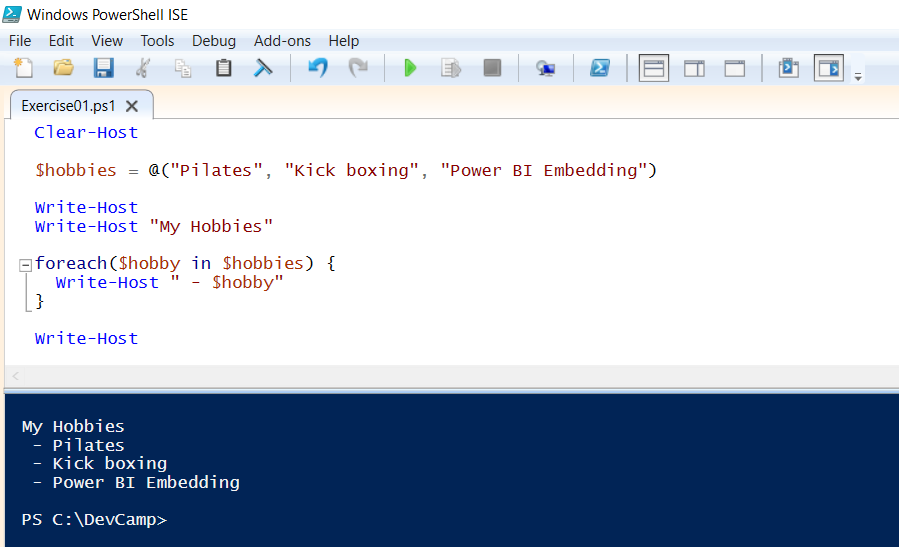
foreach($hobby in $hobbies) {

Write-Host " - $hobby"

}

Write-Host

* 1. Sss



* 1. Delete everything after the first line which calls Clear-Host
  2. Ss

$pets = @(

@{ Name="Bob"; Type="Cat" }

@{ Name="Diggity"; Type="Dog" }

@{ Name="Larry"; Type="Lizard" }

@{ Name="Penny"; Type="Porcupine" }

)

* 1. S

Write-Host

Write-Host "My Pets"

* 1. X

foreach($pet in $pets) {

$name = $pet.Name

$type = $pet.Type

Write-Host " - $name the $type"

}

* 1. Add Write-Host at the bottom.
  2. Your script should now match the following code listing.

Clear-Host

$pets = @(

@{ Name="Bob"; Type="Cat" }

@{ Name="Diggity"; Type="Dog" }

@{ Name="Larry"; Type="Lizard" }

@{ Name="Penny"; Type="Porcupine" }

)

Write-Host

Write-Host "My Pets"

foreach($pet in $pets) {

$name = $pet.Name

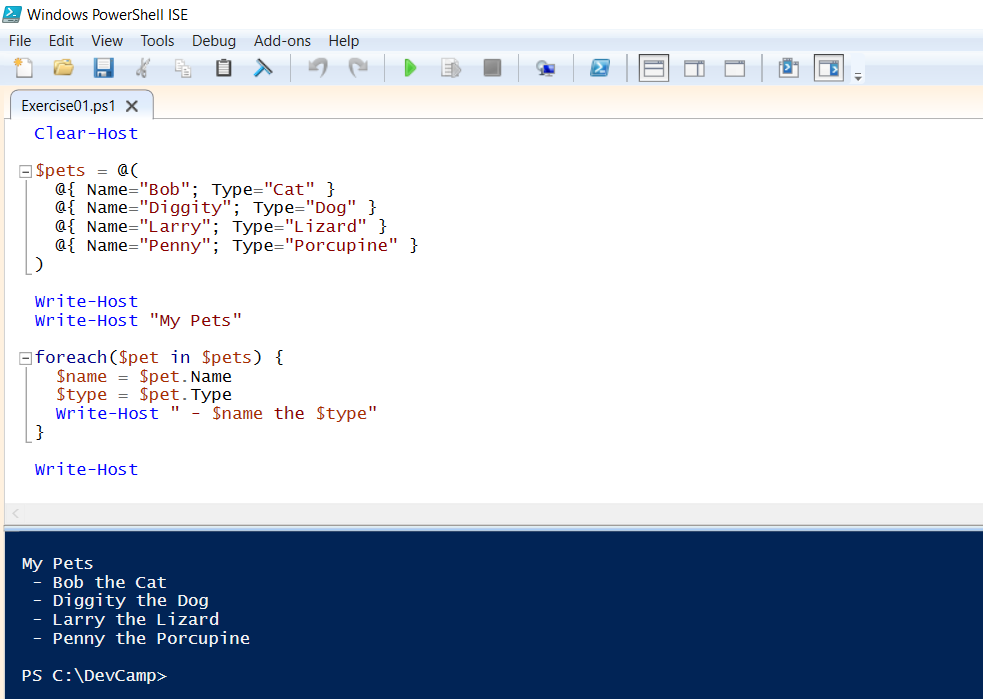
$type = $pet.Type

Write-Host " - $name the $type"

}

Write-Host

* 1. Press **{F5}** to execute the script. You should see output in the console that matches the following screenshot.



1. Write script output to a text file.
   1. Delete all the code in Exercise01.ps1.
   2. Add the following line to create a file path in the same folder as the script named **Pets.txt**.

$outputFilePath = "$PSScriptRoot/Pets.txt"

* 1. Add the following code to create an array of dictionary objects for a collection of pets.

$pets = @(

@{ Name="Bob"; Type="Cat" }

@{ Name="Diggity"; Type="Dog" }

@{ Name="Larry"; Type="Lizard" }

@{ Name="Penny"; Type="Porcupine" }

)

* 1. Add the following line of code to write a heading of **My Pets** into the output file.

"My Pets" | Out-File $outputFilePath

* 1. Create a foreach loop to enumerate the array of dictionary objects and to output a line of text for each pet which includes it name and type.

foreach($pet in $pets) {

$name = $pet.Name

$type = $pet.Type

" - $name the $type" | Out-File $outputFilePath -Append

}

* 1. Add one more line of PowerShell code to open up the text file in notepad.

notepad.exe $outputFilePath

* 1. Your script should now match the following code listing.

$outputFilePath = "$PSScriptRoot/Pets.txt"

$pets = @(

@{ Name="Bob"; Type="Cat" }

@{ Name="Diggity"; Type="Dog" }

@{ Name="Larry"; Type="Lizard" }

@{ Name="Penny"; Type="Porcupine" }

)

"My Pets" | Out-File $outputFilePath

foreach($pet in $pets) {

$name = $pet.Name

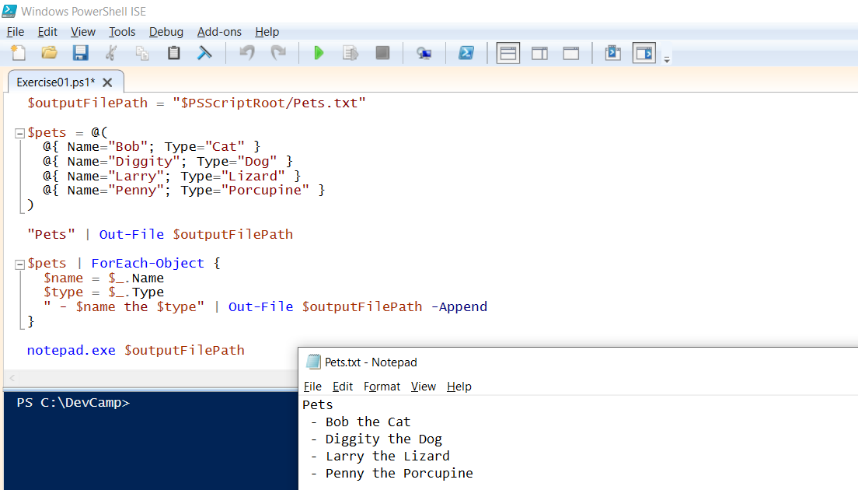
$type = $pet.Type

" - $name the $type" | Out-File $outputFilePath -Append

}

notepad.exe $outputFilePath

* 1. X



### Exercise 2: Install the Microsoft Power BI Cmdlets for Windows PowerShell

In this exercise, xxx.

1. Download the student lab files to a local folder on your developer workstation.
   1. xxxx

Install-Module -Name MicrosoftPowerBIMgmt

1. zzzzz
   1. ssss

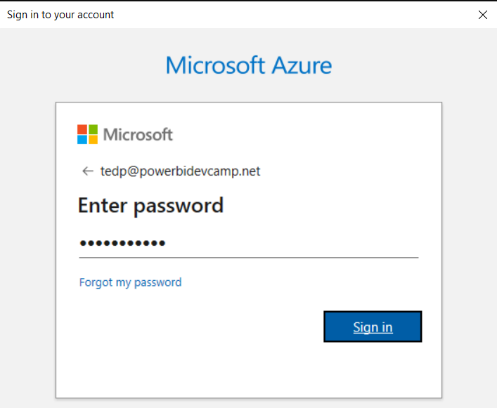
$user = Connect-PowerBIServiceAccount

$userName = $user.UserName

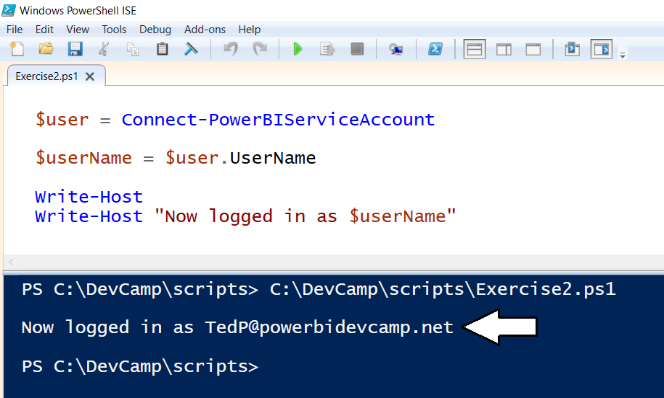
Write-Host

Write-Host "Now logged in as $userName"

1. xxx
   1. ssss



* 1. x



* 1. x

1. ssssssss
   1. sssss

# log into Azure AD user account with hard-code user name and password

$userName = "user1@tenant1.onMicrosoft.com"

$password = "myCat$rightLeg"

# convert password to secure string

$securePassword = ConvertTo-SecureString -String $password -AsPlainText -Force

# create PSCredential object to serve as login credentials

$credential = New-Object -TypeName System.Management.Automation.PSCredential `

-ArgumentList $userName, $securePassword

# log into Power BI unattended without any user interaction

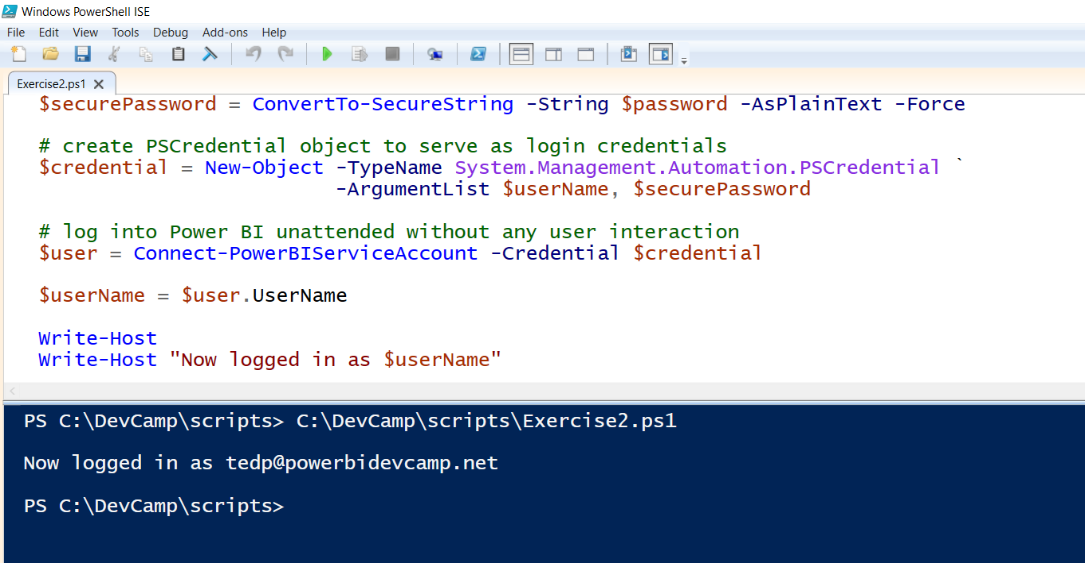
$user = Connect-PowerBIServiceAccount -Credential $credential

$userName = $user.UserName

Write-Host

Write-Host "Now logged in as $userName"

* 1. sss



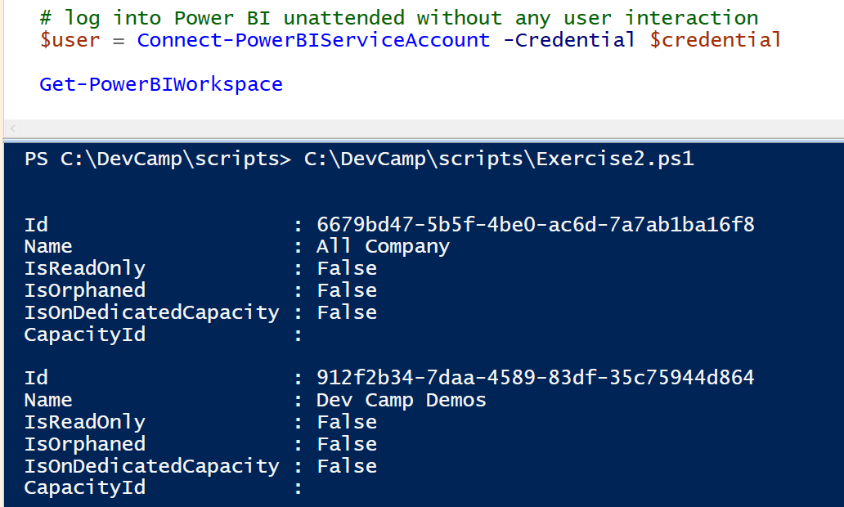
1. xxxxxx
   1. ssss

# log into Power BI unattended without any user interaction

$user = Connect-PowerBIServiceAccount -Credential $credential

Get-PowerBIWorkspace

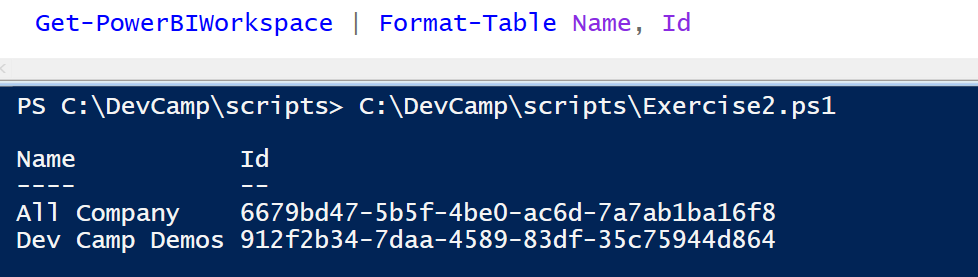
* 1. sss



* 1. sss

Get-PowerBIWorkspace | Format-Table Name, Id

* 1. zz



* 1. x

### Exercise 3: Run Power BI PowerShell Cmdlets at Organization Scope

In this exercise, xxx.

1. Download the student lab files to a local folder on your developer workstation.

### Exercise 4: Write a Script to Create Workspaces and Add Workspace Users

In this exercise, xxx.

1. Download the student lab files to a local folder on your developer workstation.
2. Create a new top-level folder

### Exercise 5: Write a Script to Upload and Publish Content

In this exercise, xxx.

1. Download the student lab files to a local folder on your developer workstation.
2. Create a new top-level folder

### Exercise 6: Write a Script to Patch Datasource Credentials

In this exercise, xxx.

1. Download the student lab files to a local folder on your developer workstation.
2. Create a new top-level folder

### Exercise 7: Write a Script to Update Dataset Parameters

In this exercise, xxx.

1. Download the student lab files to a local folder on your developer workstation.
2. Create a new top-level folder

### Exercise 8: Write a Script that Exports Power BI Activity Events

In this exercise, xxx.

1. Download the student lab files to a local folder on your developer workstation.
2. Create a new top-level folder

### Exercise 9: Write a Script that Runs as a Service Principal

In this exercise, xxx.

1. sssss
2. Takeover and Refresh a Dataset as Service Principal

### Exercise 10: Write a Script to Import a Dataflow

In this exercise, xxx.

1. Download the student lab files to a local folder on your developer workstation.
2. Create a new top-level folder