Operating Data Lake with PowerShell

Updated on: 1/13/2017

Introduction

In this lab, you will learn how to use Azure PowerShell cmdlets to interact with Azure Data Lake Analytics (ADLA) and Azure Data Lake Store (ADLS).

Prerequisites

Azure Resources

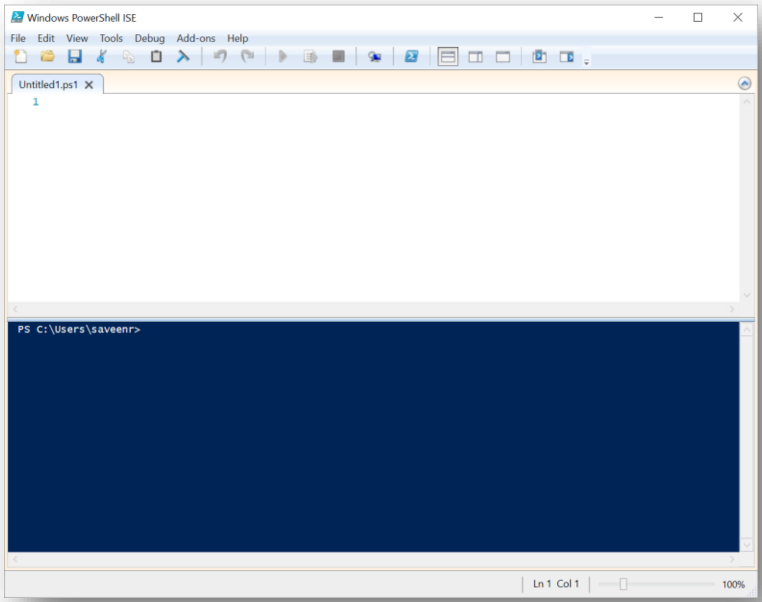
- You need an ADLA Account
- You need an ADLS Account

Your Computer

A computer running Windows with the latest version of Windows PowerShell (v5) installed and Azure PowerShell. Windows 10 ships with PowerShell v5.

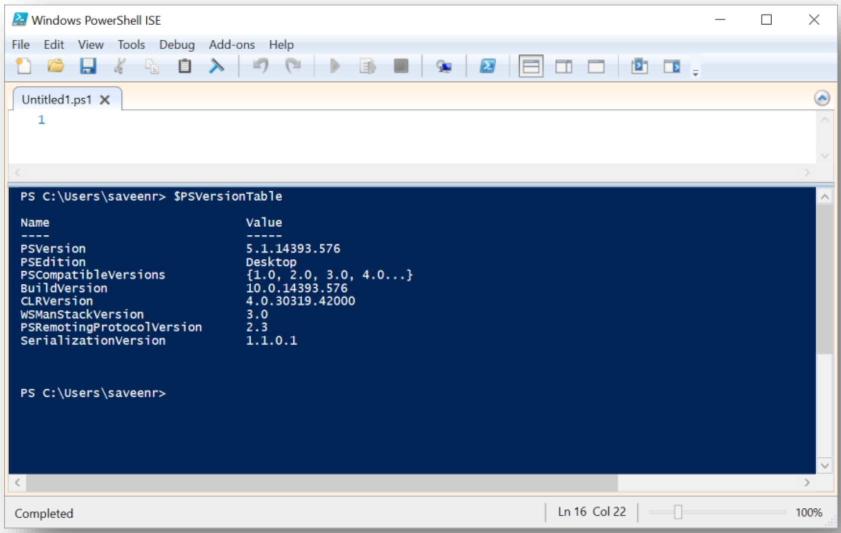
Getting Started

- First start PowerShell ISE via **Start > Windows PowerShell ISE** and launch it as an Administrator
- Verify that it looks like this:



```
PS C:\Users\saveenr>
```

- There should be a Script window and an interactive window (the blue one).
- Now check what version of powershell you have by printing out the value of $PSVersionTable



```
PS C:\Users\saveenr> $PSVersionTable

Name                           Value
----                           -----
PSVersion                      5.1.14393.576
PSEdition                      Desktop
PSCompatibleVersions           {1.0, 2.0, 3.0, 4.0...}
BuildVersion                   10.0.14393.576
CLRVersion                     4.0.30319.42000
WSManStackVersion              3.0
PSRemotingProtocolVersion      2.3
SerializationVersion           1.1.0.1


PS C:\Users\saveenr>
```

- If your **PSVersion** value does not start with 5.1 then you won't be able to complete the tasks in this document.
- We are going to be running scripts so we need to make sure we have enabled this with powershell by using this command

```
Set-ExecutionPolicy Unrestricted
```

- **IMPORTANT**: Run the Set-ExecutionPolicy command in Windows PowerShell ISE *AND* Windows PowerShell

Installing the AzureRm PowerShell module

Check if the AzureRm module is Installed

- Launch PowerShell as Administrator
- Use the command to see is AzureRm is installed

```
Get-Module -ListAvailable -Name AzureRm*
```

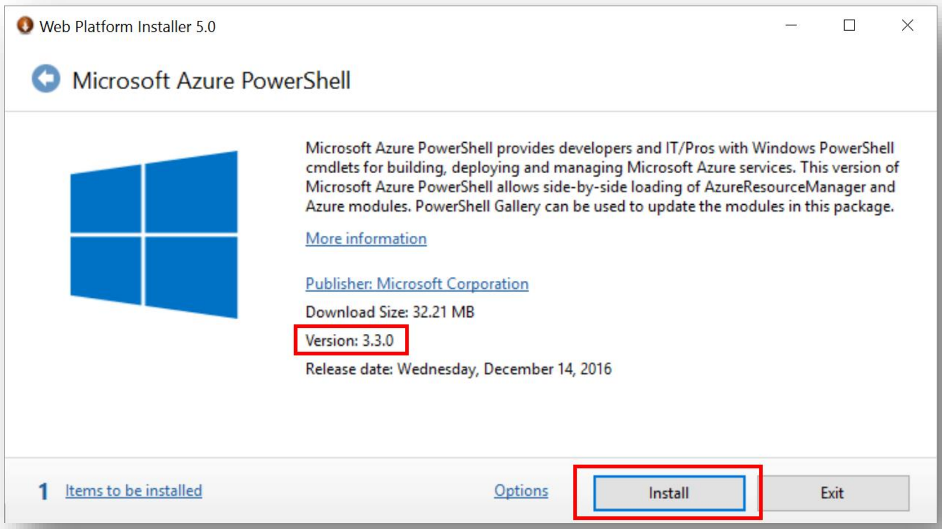
- If you don't see a lot of modules that start with "AzureRm" then AzureRm is not installed.
- To verify that AzureRm is installed

```
Import-Module AzureRm
```

Installing with the Web Platform Installer

You can also use the Web Platform Installer to get AzureRm installed on your machine.

https://www.microsoft.com/web/handlers/webpi.ashx/getinstaller/WindowsAzurePowershellGet.3f.3f.3fnew.appid



Microsoft Azure PowerShell provides developers and IT/Pros with Windows PowerShell cmdlets for building, deploying and managing Microsoft Azure services. This version of Microsoft Azure PowerShell allows side-by-side loading of AzureResourceManager and Azure modules. PowerShell Gallery can be used to update the modules in this package.

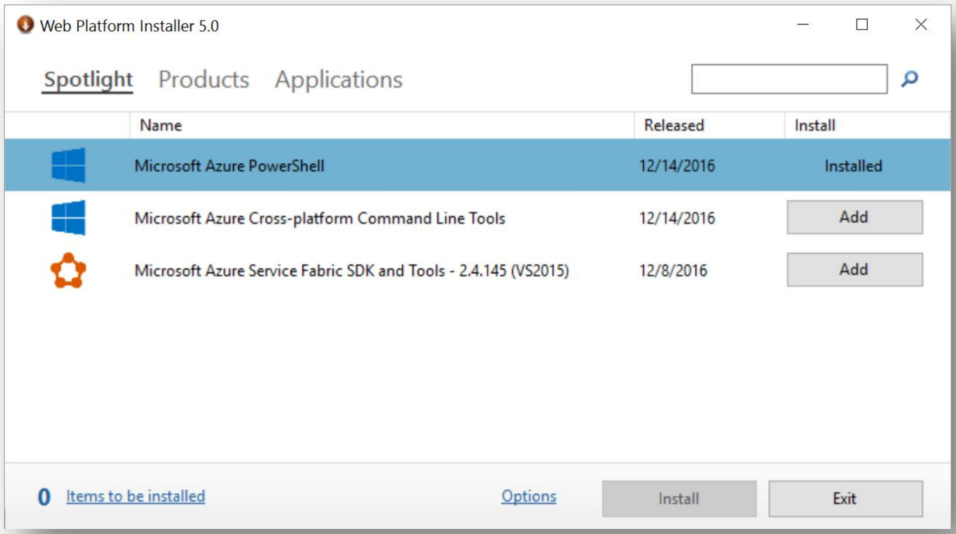
More information

Publisher: Microsoft Corporation

Download Size: 32.21 MB

Version: 3.3.0

Release date: Wednesday, December 14, 2016



Web Platform Installer 5.0

Name	Released	Install
Microsoft Azure PowerShell	12/14/2016	Installed
Microsoft Azure Cross-platform Command Line Tools	12/14/2016	Add
Microsoft Azure Service Fabric SDK and Tools - 2.4.145 (VS2015)	12/8/2016	Add

ProTip: Writing Scripts

Add this to the top of any PowerShell scripts you are writing or run them in any interactive PowerShell usage. It will simplify development and debugging.

```
Set-StrictMode -Version 2
$ErrorActionPreference = "Stop"
```

Exercise 0: Launching Azure PowerShell and Logging into Azure

- Launch PowerShell window.
- Run the following cmdlet to log in to Azure PowerShell:

```
# To login using an subscription NAME
Login-AzureRmAccount -SubscriptionName "your subscription name"

# or, to login using an subscription ID
Login-AzureRmAccount -SubscriptionId "xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx"
```

- Here's an example (don't run this)

```
Login-AzureRmAccount -SubscriptionName ADLTrainingMS
```

Exercise 1: Simplifying Logging In

Every time you run Login-AzureRmAccount as shown above you will be forced to login. This can be irritating. However, there is a way to save the login info so that you don't have keep restating which subscription, tenant, and login information to use.

- Save your login session

```
Save-AzureRmProfile -Path C:\adltraining_profile.json
```

- Close your current PowerShell window
- Launch a new PowerShell window
- Restore your previous login session by running this cmdlet

```
Select-AzureRmProfile -Path C:\adltraining_profile.json
```

Exercise 2: Exploring ADL PowerShell cmdlets

- List the ADL Analytics accounts that are available to you:

```
Get-AzureRmDataLakeAnalyticsAccount
```

- Find all the Data Lake cmdlets that you can use:

```
Get-Command *DataLake*
```

- You may notice that the commands have long object names that begin with AzureRmDataLakeAnalytics or AzureRmDataLakeStore
  - For example: Get-AzureRmDataLakeAnalyticsAccount
- These can be irritating to type, so all the cmdlets version shorter alias names. For example:
  - Actual cmdlet name: Get-AzureRmDataLakeAnalyticsAccount
  - Alias name: Get-AdlAnalyticsAccount
- We will tend to use the aliased name in this HOL because it is easier to read the code.
- To see all the aliases for ADL

```
Get-Command -commandtype alias –Name *adl*
```

- Get help on a specific cmdlet (in this case the Get-AdlAnalyticsAccount cmdlet):

```
Get-Help Get-AdlAnalyticsAccount
```

- List all Data Lake Analytics

```
Get-AdlAnalyticsAccount
```

- List all Data Lake Store accounts

```
Get-AdlStore
```

Exercise 3: Listing and submitting ADLA jobs

In this exercise you will list the jobs that have run on your ADLA account. You will also submit a new job and check its status.

List all jobs that have run on your ADLA account.

- Use the cmdlet Get-AdlJob.
- Use the -State parameter to filter by job state(s), like Running or Ended.

Submit a new job to your ADLA account.

- Save the following script to a file on your local machine, being sure to change the "UserName" part to something specific to you:

```
@searchlog =
    EXTRACT
        UserId int,
        Start DateTime,
        Region string,
        Query string,
        Duration int,
        Urls string,
        ClickedUrls string
    FROM @"/Samples/Data/SearchLog.tsv"
    USING Extractors.Tsv();

OUTPUT @searchlog
    TO @"/Samples/Output/UserName/SearchLog_TestOutput.tsv"
    USING Outputters.Tsv();
```

- Use the cmdlet **Submit-AdlJob** to submit this script with the **-ScriptPath** parameter.

Retrieve the status of the job that you submitted.

- Use the cmdlet **Get-AdlJob**.
- Pass in the job ID returned by Step 2 to the -JobId parameter.

Exercise 4: Exploring and downloading ADLS files

In this exercise, you will explore the files in your ADLS account. You will also download the file created by the job you submitted in Exercise 2.

List all files in the /Samples/Output/UserName/ folder

- Use the cmdlet **Get-AdlStoreChildItem**

Download the output of the job

- Use the cmdlet **Export-AdlStoreItem**
- Use the ADLS file path "/Samples/Output/UserName/SearchLog_TestOutput.tsv".

More Information

Once you've gone through this HOL you can try exploring the ADL PS QuickStart