1

Installing and Configuring PowerShell 7

This chapter covers the following recipes:

* Install PowerShell 7
* Using the PowerShell 7 Console
* Exploring PowerShell 7 Installation artefacts
* Exploring PowerShell 7 new features
* Building PowerShell 7 profile files
* Installing VS Code
* Installing Cascadia Code font
* Explore PSReadLine
* Installing PowerShell 7 in WSL

# Introduction

PowerShell 7 represents the latest steps in the development of PowerShell. PowerShell, first introduced to the public in 2003, was released formally as Windows PowerShell V1 in 2006. Over the next decade, Microsoft released multiple versions, ending with PowerShell 5.1. During the development of Windows PowerShell, the product moved from an add-in to Windows to be an integrated feature of Windows. Microsoft plans to support Windows PowerShell 5.;1 for a long time but no new features are likely.

The PowerShell development team began working on an Open Source version of PowerShell, based on the open-source version of .NET Core. The first two versions, PowerShell Core 6.1 and 6.2, represented a proof of concept - you can run the core functions and features of PowerShell across the Windows, Mac, and Linux platforms. But there were quite limited in terms of supporting the rich needs of the IT Pro community.

With PowerShell 7, the PowerShell team released a version of PowerShell that improved parity with Windows PowerShell. There were a few modules that did not work with PowerShell 7, and a few more that work via a compatibility mechanism described in Chapter 3. PowerShell 7.0 shipped in 2019 and has been followed by version 7.1. This book uses the term “PowerShell 7” to include both PowerShell 7.0 and 7.1. Where the discussion relates to just one specific version of PowerShell that is

Once you have installed PowerShell 7, you can run it and use it just as you used the Windows PowerShell console. The command you run to start PowerShell 7 is now pwsh.exe (versus powershell.exe for Windows PowerShell 5.1). PowerShell 7 also uses different profile file locations from Windows PowerShell. You can customize your PowerShell 7 profiles to make use of new PowerShell 7 features. You can also use different profile files for Window PowerShell and PowerShell 7.

The Windows PowerShell Integrates Scripting Environment (ISE) is a tool you use with PowerShell. The ISE, however, is not supported with PowerShell 7. To replace it, use Visual Studio Code (VS Code). VS Code is an open-source editing project that provides all the features of the ISE and great deal more.

Microsoft also developed a new font, Cascadia Code, to coincide with the launch of VS Code. This font is a nice improvement over Courier or other mono-width fonts. All screenshots of working code in this book use this new font.

PSReadline is a PowerShell module designed to provide colour-coding of PowerShell scripts in the PowerShell 7 Console. The module, included with PowerShell 7 by default, makes editing at the command line easier and more on a par with features available in Linux shells.

PowerShell 7 is cross-platform, meaning you can install it on Windows, Linux and macOS. Not every Linux distribution is supported, although many distributions, such as Ubuntu are fully supported. You can install PowerShell 7 on a native Linux host (or VM), or install it in the Windows Subsystem for Linux (WSL).

# Installing PowerShell 7

PowerShell 7 is not installed in Windows by default, at least not at the time of writing. The PowerShell team intends to have PowerShell 7 available from the Windows Store and eventually to include PowerShell 7, by default, in Windows. Until this happens, you can install PowerShell by using a simple script, Install-PowerShell.ps1 which you download from the Internet.

## Getting Ready

This recipe uses SRV1, a Windows Server 2020 workgroup host. There are no features of applications loaded on this server (yet).

## How to do it...

1. Set Execution Policy for Windows PowerShell

Set-ExecutionPolicy -ExecutionPolicy Unrestricted -Force

1. Install the latest versions of Nuget and PowerShellGet

Install-PackageProvider Nuget -MinimumVersion 2.8.5.201 -Force |

  Out-Null

Install-Module -Name PowerShellGet -Force -AllowClobber

1. Ensure the C:\Foo Folder exists

$LFHT = @{

  ItemType    = 'Directory'

  ErrorAction = 'SilentlyContinue' # should it already exist

}

New-Item -Path C:\Foo @LFHT | Out-Null

1. Download PowerShell 7 installation script

Set-Location C:\Foo

$URI = ‘https://aka.ms/install-powershell.ps1’

Invoke-RestMethod -Uri $URI |

  Out-File -FilePath C:\Foo\Install-PowerShell.ps1

1. View Installation Script Help

Get-Help -Name C:\Foo\Install-PowerShell.ps1

1. Install PowerShell 7

$EXTHT = @{

  UseMSI                 = $true

  Quiet                  = $true

  AddExplorerContextMenu = $true

  EnablePSRemoting       = $true

}

C:\Foo\Install-PowerShell.ps1 @EXTHT | Out-Null

1. For the Adventurous - install the preview and daily builds as well

C:\Foo\Install-PowerShell.ps1 -Preview -Destination C:\PWSHPreview |

Out-Null

C:\Foo\Install-PowerShell.ps1 -Daily   -Destination C:\PWSHDailBuild |

  Out-Null

1. Create Windows PowerShell default Profiles

$URI = 'https://raw.githubusercontent.com/doctordns/Wiley20/master/' +

       'Goodies/Microsoft.PowerShell\_Profile.ps1'

$ProfileFile = $Profile.CurrentUserCurrentHost

New-Item $ProfileFile -Force -WarningAction SilentlyContinue |

   Out-Null

(Invoke-WebRequest -Uri $uri -UseBasicParsing).Content |

  Out-File -FilePath  $ProfileFile

$ProfilePath = Split-Path -Path $ProfileFile

$ConsoleProfile = Join-Path -Path $ProfilePath -ChildPath 'Microsoft.PowerShell\_profile.ps1'

(Invoke-WebRequest -Uri $URI -UseBasicParsing).Content |

  Out-File -FilePath  $ConsoleProfile

1. Check versions of PowerShell 7 loaded

Get-ChildItem -Path C:\pwsh.exe -ErrorAction SilentlyContinue

## How it works...

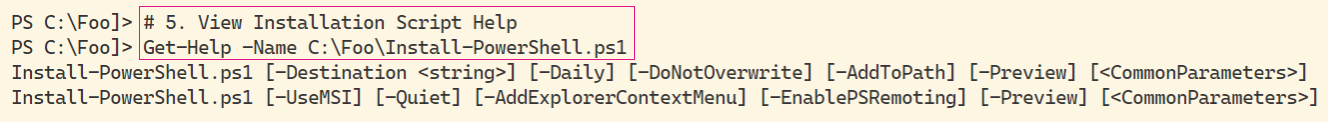
In step 1, you set the PowerShell execution policy to unrestricted which simplifies using scripts to configure hosts. In production, you may wish to set PowerShell’s execution policy to be more restrictive.

The PowerShell Gallery is a repository of PowerShell modules and scripts and is an essential resource for the IT Pro. This book makes use of several modules from the PS Gallery. In step 2, you update both the Nuget package provider (to version 2.8.5.201 or later) and an updated version of the PowerShellGet module.

Throughout this book, you use the C:\Foo folder to hold various files which you use in conjunction with the recipes. In step 3, you ensure the folder exists.

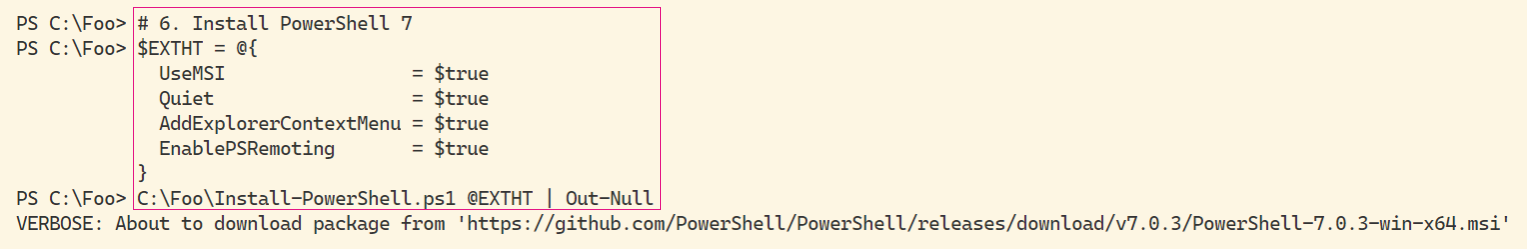
PowerShell 7 is not installed, at present, in Windows (or on in macOS or Linux), although this could change. To enable you to install PowerShell 7 in Windows, you retrieve an installation script from GitHub and store that in the C:\Foo folder. In step 4, you use a shortcut URL which points to GitHub and then use Invoke-RestMethod to download the file.

In step 5, you view the help information contained in the help file, which produces the following output:



Insert image B42024\_01\_01.png

In step 6, you use the installation script and install PowerShell 7. The commands use an MSI which you then install silently. The MSI updates the system execution path to add the PowerShell 7 installation folder. The code retrieves the latest supported version of PowerShell 7, and you can view the actual file name in the following output:



Insert image B42024\_01\_02.png

PowerShell 7 is a work in progress. Every day, the PowerShell team builds updated versions of PowerShell and releases previews of the next major release. The preview builds are mostly stable and allow you to try out new features that are coming in the next major release. The daily build allows you to view progress on a specific bug or feature. You may find it useful to install both of these (and ensure you keep them up to date as time goes by).

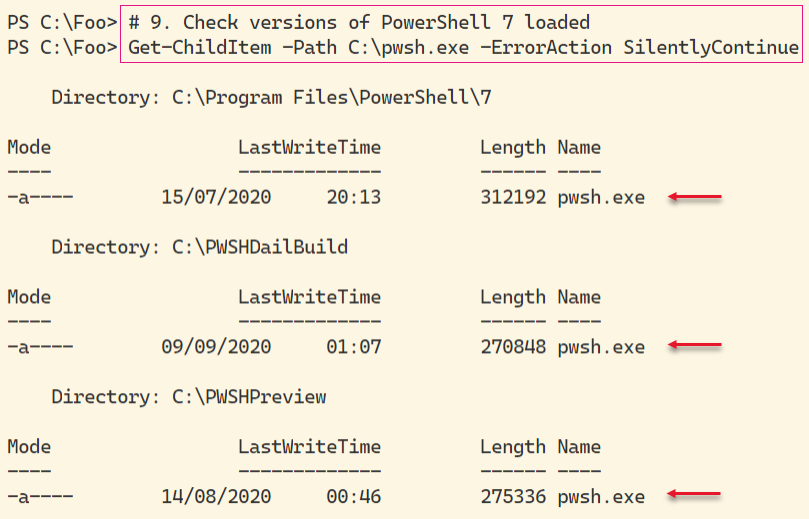
In step 7, you install the daily build and the latest Preview build, which looks like this:



Insert image B42024\_01\_03.png

PowerShell uses Profile files to enable you to configure PowerShell each time you run PowerShell (whether in the PowerShell console or as part of VS Code or the ISE). In step 8, you download a sample PowerShell profile script and save it locally. Note that the profile file you create in step 8 is for Windows PowerShell only.

The Executable name for PowerShell 8 is pwsh.exe. In step 9, you view the versions of this file as follows:



Insert image B42024\_01\_04.png

As you can see, there are 3 versions of PowerShell 7 installed on SRV1: the latest full release, the latest preview, and the build of the day.

## There's more...

In step 4, you use a shortened URL to download the Install-PowerShell.ps1 script. When you use Invoke-RestMethod, PowerShell discovers the underlying target URL for the script. The short URL allows Microsoft and the PowerShell team to publish a well-known URL and then have the flexibility to move the target location should that be necessary. The target URL, at the time of writing, is https://raw.githubusercontent.com/PowerShell/PowerShell/master/tools/install-powershell.ps1.

In step 7, you install both the latest daily build and the latest preview versions. The specific file versions you see are going to be different from the output shown here as at least for the preview versions!

# Using PWSH.EXE

With PowerShell 7, the name of the PowerShell executable is now pwsh.exe, as you saw in the previous recipe. After installing PowerShell 7 in Windows, you can start the PowerShell 7 console by clicking start and typing pwsh.exe, then hitting return. The PowerShell MSI installer does not create a start panel or taskbar shortcut.

## Getting Ready

You run this recipe on SRV1 after you have installed PowerShell 7.

## How to do it...

1. Run the PowerShell 7 console.

From the Windows desktop in SRV1, click on the Windows key, then type pwsh, followed by the enter key.

1. View the PowerShell Version

$PSVersionTable

1. View the $Host variable

$Host

1. Look at the PowerShell process

Get-Process -Id $Pid |

  Format-Custom MainModule -Depth 1

1. Look at resource usage statistics

Get-Process -Id $Pid |

  Format-List CPU,\*Memory\*

1. Update PowerShell Help

$Before = Get-Help -Name about\_\*

Update-Help -Force | Out-Null

$After = Get-Help -Name about\_\*

$Delta = $After.Count - $Before.Count

"{0} Conceptual Help Files Added" -f $Delta

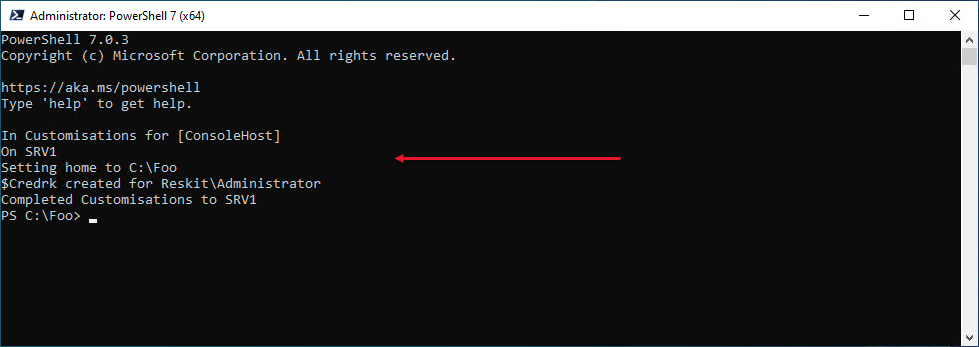
1. How many commands are available?

Get-Command |

  Group-Object -Property CommandType

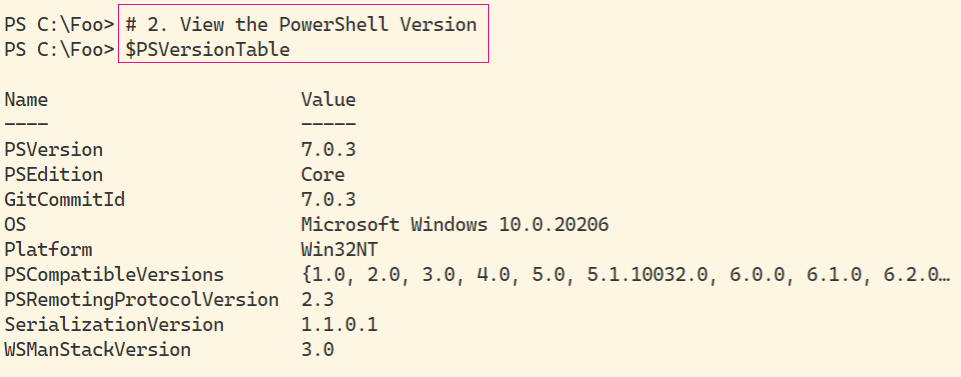
## How it works...

In step 1, you start the PowerShell 7 console on SRV1. The console should look like this:



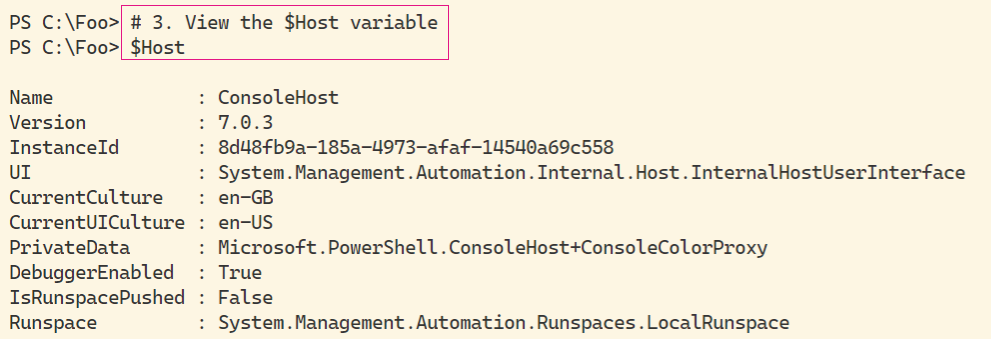
Insert image B42024\_01\_05.png

1. In step 2, you view the specific version of PowerShell by viewing the built-in variable $PSVersionTable, which looks like this:



Insert image B42024\_01\_06.png

In step 3, you examine the $Host variable to determine details about the PowerShell 7 host (the PowerShell console), which looks like this:



Insert image B42024\_01\_07.png

As you can see, in this case, you can see, the current culture is EN-GB. You may see a different value depending on which specific version of Windows Server you are using.

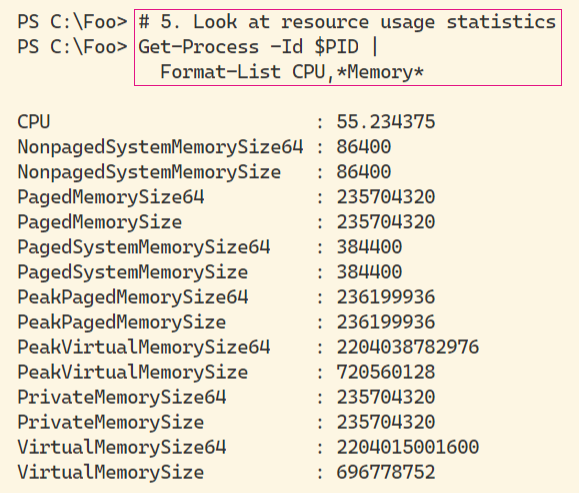
In step 4, you use Get-Process to look at the details of the PowerShell process, which looks like this:



Insert image B42024\_01\_08.png

In this figure, you can see the path to the PowerShell 7 executable. This value changes if you are running the release version or the daily/preview releases.

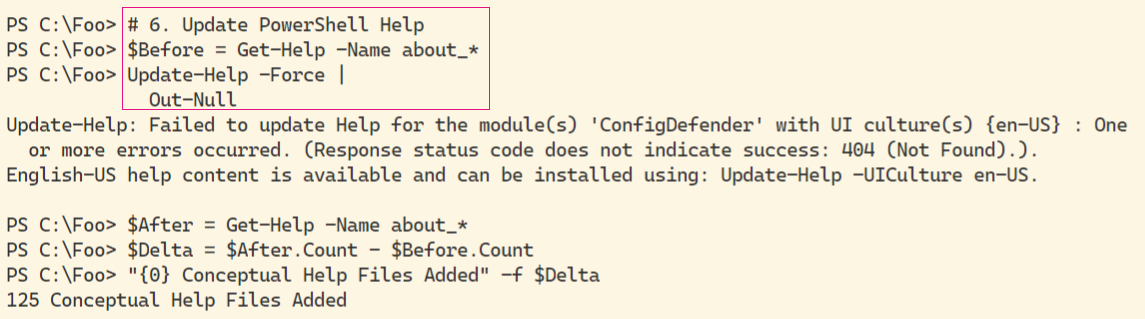
You can see, in step 5, details of resource usage of the pwsh.exe process running on the SRV1 host.



Insert image B42024\_01\_09.png

The values of each of the performance counters are likely to. You may see different values.

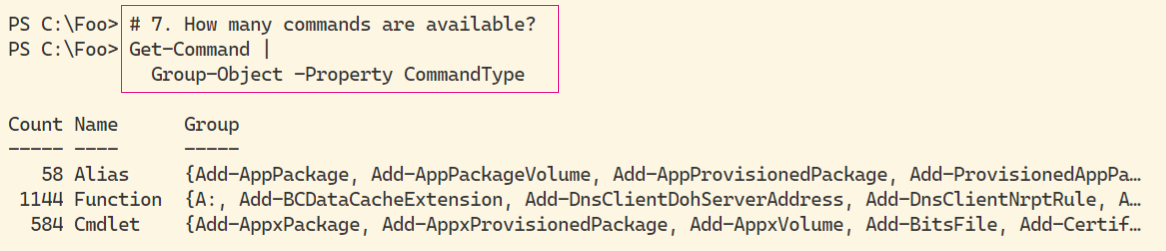
By default, Powershell 7, like Windows PowerShell, ships with minimum help files. You can, as you can see in step 6, use the Update-Help command to download updated PowerShell help content, like this:



Insert image B42024\_01\_10.png

As you can see from the output, not all help files were updated. In this case, the ConfigDefender module does not, at present, have updated help information. Also note that although the UK English versions of help details may be missing, there are US English versions that you can install which may be useful.

Commands, in PowerShell, include functions, cmdlets, and aliases. In step 7, you examine how many of each time of command is available by default, like this:



Insert image B42024\_01\_11.png

## There's more...

In step 1, you open the PowerShell console for the version of PowerShell you installed in “Installing PowerShell 7”. At the release of PowerShell 7.1, the version number you would see is 7.1.0. By the time you read this book, that version number may have advanced. To ensure you have the latest released version of PowerShell 7, re-run the Install-PowerShell.ps1 script you downloaded in “Installing PowerShell 7” In this step, you can also see the output generated by the Write-Host statements in the profile file you set up in “Introducing PowerShell 7”.

In step 4, you use the variable $PID, which contains the Windows process identifier of the PowerShell 7 console process. The actual value of $PID changes each time you run PowerShell, but the value always contains the process id of the current console process.

In step 7, you saw that you had 1786 commands available. This number changes as you add more features (and their accompanying modules) or download and install modules from repositories such as the PowerShell Gallery.

# Exploring PowerShell 7 Installation Artifacts

In PowerShell 7, certain objects added by the PowerShell 7 installer (and PowerShell 7) differ from those used by Windows PowerShell.

## Getting Ready

This recipe uses SRV1 after you have installed PowerShell 7.

## How to do it...

1. Step by step with code:

Get-Package -ProviderName 'msu' |

Select-Object -ExpandProperty Name

1 2 3 4 5 6 7

1234567890123456789012345678901234567890123456789012345678901234567890123

This shows how that the line width for code will be 73 characters.

## How it works...

## There's more...

1. Some things of interest in this recipe

# Building PowerShell 7 profile files

In Windows PowerShell and PowerShell 7, profile files contain code which PowerShell runs each time you start a new PowerSHell console or a new instance of the ISE (or VS Code). These files enable you to pre-configure a PowerShell 7. You can add variables, PowerShell PS drives, functions and more using Profiles. As part of this book, there are PowerShell (and VS Code) profile file samples which you can download from GitHub.

This recipe download and installs the profile files for PowerShell 7.

## Getting Ready

You run this recipe on SRV1 after you have installed PowerShell 7.

## How to do it...

1. Step by step with code:

Get-Package -ProviderName 'msu' |

Select-Object -ExpandProperty Name

1 2 3 4 5 6 7

1234567890123456789012345678901234567890123456789012345678901234567890123

This shows how that the line width for code will be 73 characters.

## How it works...

## There's more...

# Installing VS Code

The Windows PowerShell ISE was a great too, first introduced with Windows PowerShell v2 and vastly improved with v3. Sadly, this tool has reaced feature completeness and Microsoft has no plans for further development. In its place, however, is Visual Studio Code, or VS Code. This open-source tool provides a large range of features for IT pros and others. For IT Pros, this should be your editor of choice.

For more details on VS Code, see: https://code.visualstudio.com/.

## Getting Ready

You run this recipe on SRV1 after you have installed PowerShell 7 and have created a console profile file.

## How to do it...

1. Step by step with code:

Get-Package -ProviderName 'msu' |

Select-Object -ExpandProperty Name

1 2 3 4 5 6 7

1234567890123456789012345678901234567890123456789012345678901234567890123

This shows how that the line width for code will be 73 characters.

## How it works...

1. Screen shots for each step that generates one

## There's more...

1. Some things of interest in this recipe

# Installing Cascadia Code Font

This recipe, blah blah

## Getting Ready

Specific stuff you need to do this recipe

## How to do it...

1. Step by step with code:

Get-Package -ProviderName 'msu' |

Select-Object -ExpandProperty Name

1 2 3 4 5 6 7

1234567890123456789012345678901234567890123456789012345678901234567890123

This shows how that the line width for code will be 73 characters.

## How it works...

## There's more...

# Exploring PSReadLine

This recipe, blah blah

## Getting Ready

Specific stuff you need to do this recipe

## How to do it...

1. Step by step with code:

Get-Package -ProviderName 'msu' |

Select-Object -ExpandProperty Name

1 2 3 4 5 6 7

1234567890123456789012345678901234567890123456789012345678901234567890123

This shows how that the line width for code will be 73 characters.

## How it works...

1. Screen shots for each step that generates one

## There's more...

1. Some things of interest in this recipe

# Installing PowerShell 7 in Windows Subsystem for LInux

PowerShell 7 is cross-platform and you can use PowerShell on Windows, macOS and most distributions of Linux. It is a feature of Linux that the installations of applicaioons such as PowerShell differs. The tools you use for example in Ubuntu, are different from those you use in other distributions.

With the latest versions of WInedows and Windows Server, Microsoft has implemented the Windows Subsystem for Linux (WSL). WSL allows you to run a (supported) Linux distribution inside Windows. For a deeper look at WSL, see: https://docs.microsoft.com/en-us/windows/wsl/about.

## Getting Ready

Specific stuff you need to do this recipe

## How to do it...

1. Step by step with code:

Get-Package -ProviderName 'msu' |

Select-Object -ExpandProperty Name

1 2 3 4 5 6 7

1234567890123456789012345678901234567890123456789012345678901234567890123

This shows how that the line width for code will be 73 characters.

## How it works...

## There's more...