C# myPowerShell

Knowledge is a process of piling up facts; wisdom lies in their simplification.
Martin H Fischer

Modification Record

Issue	Date	Author	Changes (Including the change authority)
A	19 July 2021	ZizWiz	Original

Contents

Modification Record **2**

1. Introduction **4**

2. What is Powershell **4**

3. What is a cmdlet **4**
3.1. Alias 4

4. Cmdlet help **5**

5. The myPowershell app **5**
5.1. Nuget Packages required. 5
5.2. The UI 5
5.3. Scripts 6

6. How to unblock files **8**

7. Check it in Windows Sandbox **9**
7.1. Properties 9
7.2. **Prerequisites** 9
7.3. How to set up Windows Sandbox 10

8. Appendix – List of Commands **11**

9. Alias **17**

1. Introduction

For me it all started when I got code down from GitHub and could not compile it. When you download from the cloud Win10 puts a flag on some types of files like .resx that blocks them. This is picked up by VS and it will not allow the code to compile. I then had to manually go through the files unblocking each as required. I looked for a method I could use in an app to do this for me. I then saw that I could do this in Powershell, this is the story of how I achieved that.

2. What is Powershell

Windows PowerShell 1.0 was launched in November 2006. It is a powerful command line shell for Windows. On Unix/Linux you have shells such as *csh* and *bash*. Powershell in some ways has more capabilities than them as it can read and write objects, as opposed to conventional shells that can only process strings of text. PowerShell runs on the .NET platform, the objects that are used are .NET objects, which makes it an ideal scripting language for .NET programs.

3. What is a cmdlet

A cmdlet or "Command let" is a lightweight command used in the Windows PowerShell environment. The Windows PowerShell runtime invokes these cmdlets at command prompt. You can create and invoke them programmatically through Windows PowerShell APIs. An example of a cmdlet is `Get-ChildItem`.

3.1. Alias

These are even shorter terms that you can associate with the cmdlet. In early versions of Powershell there were not many inbuilt aliases, so you had to tell Powershell what you had chosen. Today there are many inbuild aliases and the choice of using the cmdlet or its alias is up to the user. Many aliases have been chosen for those more familiar with Windows batch commands and some for those more familiar with Linux. You can as mentioned earlier still create your own.

e.g. The cmdlet `Get-ChildItem` can be shortened to its alias `dir` which if you are a windows user you may be more familiar with or `ls` if you are more familiar with Unix/Linux commands as we see in the following:

- `Get-ChildItem -Path “directory path” -Recurse | Unblock-File`

Can be written as

- `dir -Path “directory path” -Recurse | Unblock-File`

or

- `ls -Path “directory path” -Recurse | Unblock-File`

4. Cmdlet help

I found that when I asked Powershell to supply help for a particular cmdlet that there was nothing available. This is strange as there should be and then I found that you can update the help for the cmdlets by using the following command

- `Update-Help`

After this update I was able to get help for the cmdlet. It may be wise to occasionally just update your help.

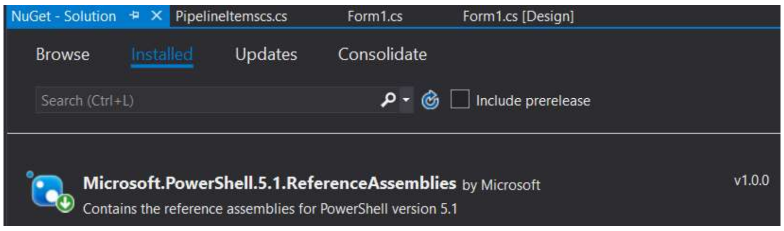
5. The myPowershell app

I decided that I would investigate making an app to help me run Powershell commands. By doing this it would give me a better understanding of the deeper level commands and structures. As with a lot of apps today I found someone had started to make one and I used this as the basis for the one I would create. The original can be found at https://www.codeproject.com/Articles/18409/Asynchronously-Execute-PowerShell-Scripts-from-C

I started my app from the beginning but did make the UI similar as I liked it and I did use some code from the original. I did find that the original was made with .Net v2.0 and I was using .Net v4.8. This means that many items do not work the same way. By the time I had worked out how to do a specific topic in v4.8 I found had to write the original in a different way so much of the code is written by me while studying how the original was written. Once I had the basics working, I started to look at making the app work the way I wanted it to work these will be detailed later.

5.1. Nuget Packages required.

The first thing I found I need was the Powershell Reference assemblies and luckily this is available as a Nuget package which I installed.



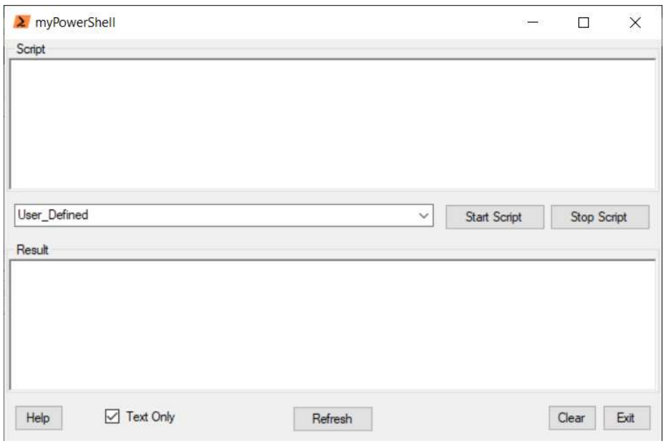
Microsoft.PowerShell.5.1.ReferenceAssemblies by Microsoft v1.0.0
Contains the reference assemblies for PowerShell version 5.1

5.2. The UI

I wanted to make it look similar to the original, so I have two windows. One the shows the script and another the result. The result window can change from a RickTextBox to a GridView depending on what is likely to be returned. I am still working on understanding how to format what is returned.

I wanted to pick up the list of scripts I could use from a folder. This will allow the user to write scripts and save them for reuse. If you do add new scripts to the folder then you can either restart the app to pick them up or click on the "refresh" button. The refresh button will clear both windows and reinitialise the list from the script folder.

The Text Only tick box is explained later.



In the original I saw that you could drag in a file and drop it into the Script window. I have added this feature to my app. It will allow you to drag a txt file and then will parse the text file and write the contents into the Script window.

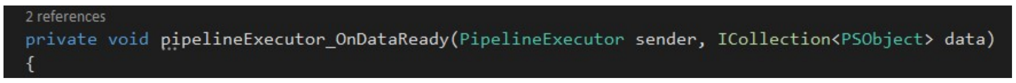
The drop down with the list of scripts in will always start on "User_Defined". When in this mode you will be able to write into the script window from the keyboard your own script and then press the "Start Script" button to run it.

5.3. Scripts

The script naming convention is important. I use the following but will expand it as required. The name has a prefix, reading it LtoR the first part before the underscore.

Prefix	Meaning
config	Used to configure the Powershell environment.
info	These mostly bring back info about the PC you are on.
operators	Demonstrate how to use some mathematical operators.
prog	Demonstrate how to use some programming features like loops.
util	Utilities which you may write to do a specific task.

This prefix is used by the app to determine how the result should be displayed. I am still learning how to do this section but so far, I know that those prefixed with "info" will need to use the data grid as there return is a PSObject containing many PSPropertyInfo. It is the property info that we will want to show in the Gridview and not the richtextbox so we need to switch the one that is shown and use. We do this in the function

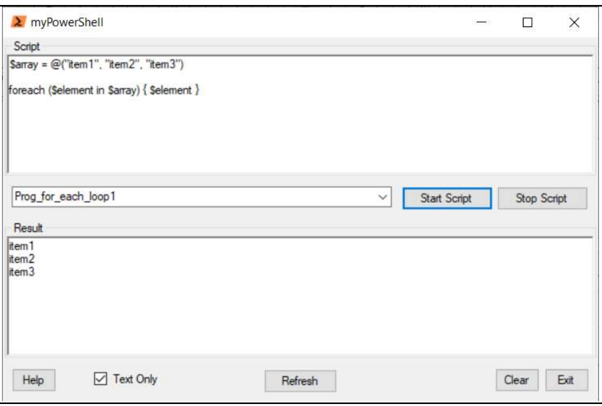
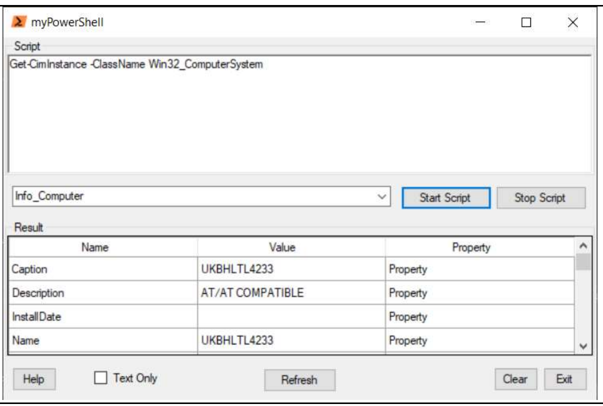


```
2 references
private void pipelineExecutor_OnDataReady(PipelineExecutor sender, ICollection<PSObject> data)
{
```

You will also note the Tickbox called “Text Only”. When Ticked the returning info will be displayed in a RichTextBox and when unticked it will be displayed in the Gridview.

If I need more ways to display the data in the future then I will add more choices here. When you choose a script with Info prefix then the box is automatically unticked and when you choose another then the box will be ticked. When using “User_Defined” scripts you will need to do this manually for your script.

The results can look as follows:

RichTextBox	GridView
	

```
$array = @("item1", "item2", "item3")

foreach ($element in $array) { $element }
```

```
item1
item2
item3
```

```
Get-CimInstance -ClassName Win32_ComputerSystem
```

Name	Value	Property
Caption	UKBHLTL4233	Property
Description	AT/AT COMPATIBLE	Property
InstallDate		Property
Name	UKBHLTL4233	Property

6. How to unblock files

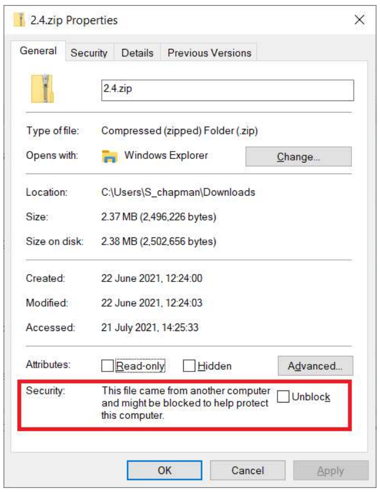
This is the reason why I started creating this app. If you have ever downloaded some files and then when you go to use the file you get a warning message like

```
This file came from another computer and might be blocked to help protect this computer.
```

Or in the case of Visual Studio it will not allow you to compile the code.

You can get round the issue but only do this if you trust the file and be careful if using a PC controlled by a company as their policies may say do not use the file. In that case if you are allowed you can use the Sandbox to run it as explained later.

To unblock the file right-click on a single file and choose Properties, you can see the Security section at the bottom of the window. If you do this on a zip file, then all the items inside also get unblocked. At the bottom of the box, you will see as highlighted in red below where you can choose to unblock the file.



2.4.zip Properties

General | Security | Details | Previous Versions

2.4.zip

Type of file:	Compressed (zipped) Folder (.zip)
Opens with:	Windows Explorer
Location:	C:\Users\S_chapman\Downloads
Size:	2.37 MB (2,496,226 bytes)
Size on disk:	2.38 MB (2,502,656 bytes)
Created:	22 June 2021, 12:24:00
Modified:	22 June 2021, 12:24:03
Accessed:	21 July 2021, 14:25:33
Attributes:	☐ Read-only ☐ Hidden
Security:	This file came from another computer and might be blocked to help protect this computer. ☐ Unblock

e.g. The cmdlet `Get-ChildItem` can be shortened to its alias `dir` which if you are a windows user you may be more familiar with or `ls` if you are more familiar with Unix/Linux commands as we see in the following:

- `Get-ChildItem -Path “directory path” -Recurse | Unblock-File`

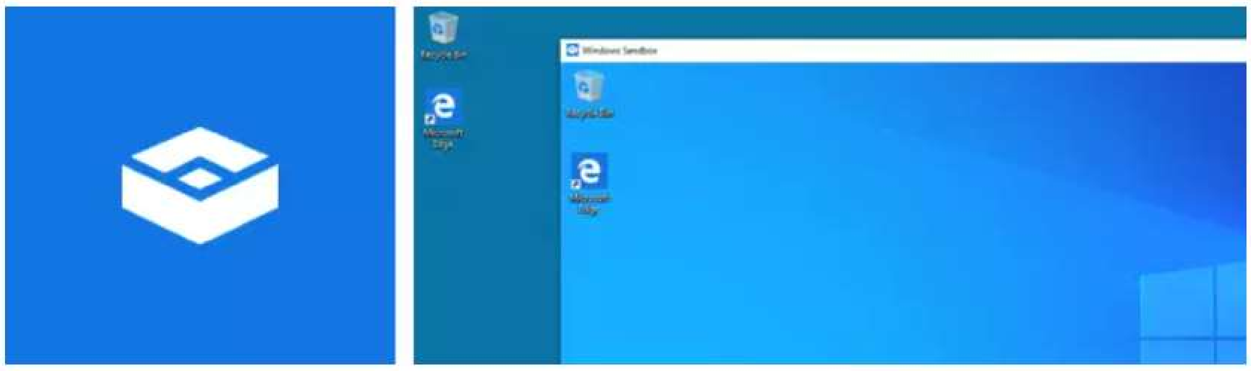
Can be written as

- `dir -Path “directory path” -Recurse | Unblock-File`

or

- `ls -Path “directory path” -Recurse | Unblock-File`

7. Check it in Windows Sandbox



Windows Sandbox is a lightweight desktop environment tailored for safely running applications in isolation from the rest of your system. Windows Sandbox provides an isolated, temporary, desktop environment where you can run untrusted software without the fear of lasting impact to your PC. Any software installed in Windows Sandbox stays only in the sandbox and cannot affect your host. Once Windows Sandbox is closed, all the software with all its files and state are permanently deleted. Unfortunately, Windows Sandbox is switched off by default and later we will explain how to switch it on to be able to use it.

7.1. Properties

Microsoft explain that Windows Sandbox has the following properties:

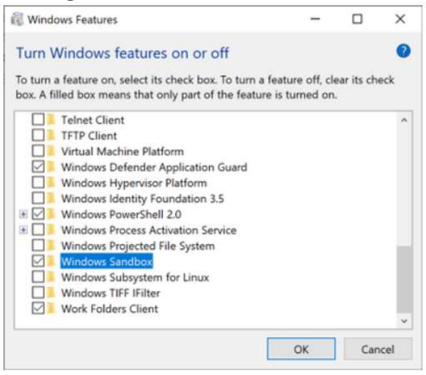
1. Part of Windows – everything required for this feature ships with Windows 10 Pro and Enterprise.
2. Pristine – every time Windows Sandbox runs, it's as clean as a brand-new installation of Windows
3. Disposable – nothing persists on the device; everything is discarded after you close the application
4. Secure – uses hardware-based virtualisation for kernel isolation, which relies on the Microsoft's hypervisor to run a separate kernel which isolates Windows Sandbox from the host
5. Efficient – uses integrated kernel scheduler, smart memory management, and virtual GPU

7.2. **Prerequisites**

1. A 64-bit processor capable of virtualization, with at least two CPU cores; Microsoft recommends a quad-core chip.
2. Windows Pro, Enterprise, or Server
3. At least 4GB of RAM (8GB recommended)
4. At least 1GB of free disk space (SSD recommended)
5. Virtualization capabilities enabled in BIOS

7.3. How to set up Windows Sandbox

- Install Windows 10 Pro or Enterprise, Insider build 18305 or newer
- Enable virtualization:
    a. If you are using a physical machine, ensure virtualization capabilities are enabled in the BIOS.
    b. If you are using a virtual machine, enable nested virtualization with this PowerShell cmdlet:
        i. `Set-VMProcessor -VMName <VMName> -ExposeVirtualizationExtensions $true`
- Open Windows Features, and then select Windows Sandbox. Select **OK** to install Windows Sandbox. You might be asked to restart the computer.



Windows Features

Turn Windows features on or off

To turn a feature on, select its check box. To turn a feature off, clear its check box. A filled box means that only part of the feature is turned on.

- [ ] Telnet Client
- [ ] TFTP Client
- [ ] Virtual Machine Platform
- [✓] Windows Defender Application Guard
- [ ] Windows Hypervisor Platform
- [ ] Windows Identity Foundation 3.5
- [✓] Windows PowerShell 2.0
- [ ] Windows Process Activation Service
- [ ] Windows Projected File System
- [✓] Windows Sandbox
- [ ] Windows Subsystem for Linux
- [ ] Windows TIFF IFilter
- [✓] Work Folders Client

- Using the Start menu, find Windows Sandbox, run it and allow the elevation
- Copy an executable file from the host
- Paste the executable file in the window of Windows Sandbox (on the Windows desktop)
- Run the executable in the Windows Sandbox; if it is an installer go ahead and install it
- Run the application and use it as you normally do
- When you're done experimenting, you can simply close the Windows Sandbox application. All sandbox content will be discarded and permanently deleted
- Confirm that the host does not have any of the modifications that you made in Windows Sandbox.

8. Appendix – List of Commands

PowerShell cmdlet name, and a description of what the command actually does.

Command alias	**Cmdlet name**	**Description of command**
%	ForEach-Object	Performs an operation against each item in a collection of input objects.
?	Where-Object	Selects objects from a collection based on their property values.
ac	Add-Content	Appends content, such as words or data, to a file.
asnp	Add-PSSnapIn	Adds one or more Windows PowerShell snap-ins to the current session.
cat	Get-Content	Gets the contents of a file.
cd	Set-Location	Sets the current working location to a specified location.
chdir	Set-Location	Sets the current working location to a specified location.
clc	Clear-Content	Deletes the contents of an item, but does not delete the item.
clear	Clear-Host	Clears the display in the host program.
clhy	Clear-History	Deletes entries from the command history.
cli	Clear-Item	Deletes the contents of an item, but does not delete the item.
clp	Clear-ItemProperty	Deletes the value of a property but does not delete the property.
cls	Clear-Host	Clears the display in the host program.
clv	Clear-Variable	Deletes the value of a variable.
cnsn	Connect-PSSession	Reconnects to disconnected sessions
compare	Compare-Object	Compares two sets of objects.
copy	Copy-Item	Copies an item from one location to another.
cp	Copy-Item	Copies an item from one location to another.
cpi	Copy-Item	Copies an item from one location to another.
cpp	Copy-ItemProperty	Copies a property and value from a specified location to another location.
curl	Invoke-WebRequest	Gets content from a webpage on the Internet.

cvpa	Convert-Path	Converts a path from a Windows PowerShell path to a Windows PowerShell provider path.
dbp	Disable-PSBreakpoint	Disables the breakpoints in the current console.
del	Remove-Item	Deletes files and folders.
diff	Compare-Object	Compares two sets of objects.
dir	Get-ChildItem	Gets the files and folders in a file system drive.
dnsn	Disconnect-PSSession	Disconnects from a session.
ebp	Enable-PSBreakpoint	Enables the breakpoints in the current console.
echo	Write-Output	Sends the specified objects to the next command in the pipeline. If the command is the last command in the pipeline, the objects are displayed in the console.
epal	Export-Alias	Exports information about currently defined aliases to a file.
epcsv	Export-Csv	Converts objects into a series of comma-separated (CSV) strings and saves the strings in a CSV file.
epsn	Export-PSSession	Imports commands from another session and saves them in a Windows PowerShell module.
erase	Remove-Item	Deletes files and folders.
etsn	Enter-PSSession	Starts an interactive session with a remote computer.
exsn	Exit-PSSession	Ends an interactive session with a remote computer.
fc	Format-Custom	Uses a customized view to format the output.
fl	Format-List	Formats the output as a list of properties in which each property appears on a new line.
foreach	ForEach-Object	Performs an operation against each item in a collection of input objects.
ft	Format-Table	Formats the output as a table.
fw	Format-Wide	Formats objects as a wide table that displays only one property of each object.
gal	Get-Alias	Gets the aliases for the current session.
gbp	Get-PSBreakpoint	Gets the breakpoints that are set in the current session.
gc	Get-Content	Gets the contents of a file.
gci	Get-ChildItem	Gets the files and folders in a file system drive.
gcm	Get-Command	Gets all commands.

gcs	Get-PSCallStack	Displays the current call stack.
gdr	Get-PSDrive	Gets drives in the current session.
ghy	Get-History	Gets a list of the commands entered during the current session.
gi	Get-Item	Gets files and folders.
gjb	Get-Job	Gets Windows PowerShell background jobs that are running in the current session.
gl	Get-Location	Gets information about the current working location or a location stack.
gm	Get-Member	Gets the properties and methods of objects.
gmo	Get-Module	Gets the modules that have been imported or that can be imported into the current session.
gp	Get-ItemProperty	Gets the properties of a specified item.
gps	Get-Process	Gets the processes that are running on the local computer or a remote computer.
group	Group-Object	Groups objects that contain the same value for specified properties.
gsn	Get-PSSession	Gets the Windows PowerShell sessions on local and remote computers.
gsnp	Get-PSSnapIn	Gets the Windows PowerShell snap-ins on the computer.
gsv	Get-Service	Gets the services on a local or remote computer.
gu	Get-Unique	Returns unique items from a sorted list.
gv	Get-Variable	Gets the variables in the current console.
gwmi	Get-WmiObject	Gets instances of Windows Management Instrumentation (WMI) classes or information about the available classes.
h	Get-History	Gets a list of the commands entered during the current session.
history	Get-History	Gets a list of the commands entered during the current session.
icm	Invoke-Command	Runs commands on local and remote computers.
iex	Invoke-Expression	Runs commands or expressions on the local computer.
ihy	Invoke-History	Runs commands from the session history.

ii	Invoke-Item	Performs the default action on the specified item.
ipal	Import-Alias	Imports an alias list from a file.
ipcsv	Import-Csv	Creates table-like custom objects from the items in a CSV file.
ipmo	Import-Module	Adds modules to the current session.
ipsn	Import-PSSes sion	Imports commands from another session into the current session.
irm	Invoke-RestMethod	Sends an HTTP or HTTPS request to a RESTful web service.
ise	powershell_ise.exe	Explains how to use the PowerShell_ISE.exe command-line tool.
iwmi	Invoke-WMIMethod	Calls Windows Management Instrumentation (WMI) methods.
iwr	Invoke-WebRequest	Gets content from a web page on the Internet.
kill	Stop-Process	Stops one or more running processes.
lp	Out-Printer	Sends output to a printer.
ls	Get-ChildItem	Gets the files and folders in a file system drive.
man	help	Displays information about Windows PowerShell commands and concepts.
md	mkdir	Creates a new item.
measure	Measure-Object	Calculates the numeric properties of objects, and the characters, words, and lines in string objects, such as files of text.
mi	Move-Item	Moves an item from one location to another.
mount	New-PSDrive	Creates temporary and persistent mapped network drives.
move	Move-Item	Moves an item from one location to another.
mp	Move-ItemProperty	Moves a property from one location to another.
mv	Move-Item	Moves an item from one location to another.
nal	New-Alias	Creates a new alias.
ndr	New-PSDrive	Creates temporary and persistent mapped network drives.
ni	New-Item	Creates a new item.
nmo	New-Module	Creates a new dynamic module that exists only in memory.

npssc	New-PSSessionConfigurationFile	Creates a file that defines a session configuration.
nsn	New-PSSession	Creates a persistent connection to a local or remote computer.
nv	New-Variable	Creates a new variable.
ogv	Out-GridView	Sends output to an interactive table in a separate window.
oh	Out-Host	Sends output to the command line.
popd	Pop-Location	Changes the current location to the location most recently pushed to the stack. You can pop the location from the default stack or from a stack that you create by using the Push-Location cmdlet.
ps	Get-Process	Gets the processes that are running on the local computer or a remote computer.
pushd	Push-Location	Adds the current location to the top of a location stack.
pwd	Get-Location	Gets information about the current working location or a location stack.
r	Invoke-History	Runs commands from the session history.
rbp	Remove-PSBreakpoint	Deletes breakpoints from the current console.
rcjb	Receive-Job	Gets the results of the Windows PowerShell background jobs in the current session.
rcsn	Receive-PSSession	Gets results of commands in disconnected sessions.
rd	Remove-Item	Deletes files and folders.
rdr	Remove-PSDrive	Deletes temporary Windows PowerShell drives and disconnects mapped network drives.
ren	Rename-Item	Renames an item in a Windows PowerShell provider namespace.
ri	Remove-Item	Deletes files and folders.
rjb	Remove-Job	Deletes a Windows PowerShell background job.
rm	Remove-Item	Deletes files and folders.
rmdir	Remove-Item	Deletes files and folders.
rmo	Remove-Module	Removes modules from the current session.
rni	Rename-Item	Renames an item in a Windows PowerShell provider namespace.

rnp	Rename-ItemProperty	Renames a property of an item.
rp	Remove-ItemProperty	Deletes the property and its value from an item.
rsn	Remove-PSSession	Closes one or more Windows PowerShell sessions (PSSessions).
rsnp	Remove-PSSnapin	Removes Windows PowerShell snap-ins from the current session.
rujb	Resume-Job	Restarts a suspended job
rv	Remove-Variable	Deletes a variable and its value.
rvpa	Resolve-Path	Resolves the wildcard characters in a path and displays the path contents.
rwmi	Remove-WMIObject	Deletes an instance of an existing Windows Management Instrumentation (WMI) class.
sajb	Start-Job	Starts a Windows PowerShell background job.
sal	Set-Alias	Creates or changes an alias (alternate name) for a cmdlet or other command element in the current Windows PowerShell session.
saps	Start-Process	Starts one or more processes on the local computer.
sasv	Start-Service	Starts one or more stopped services.
sbp	Set-PSBreakpoint	Sets a breakpoint on a line, command, or variable.
sc	Set-Content	Replaces the contents of a file with contents that you specify.
select	Select-Object	Selects objects or object properties.
set	Set-Variable	Sets the value of a variable. Creates the variable if one with the requested name does not exist.
shcm	Show-Command	Creates Windows PowerShell commands in a graphical command window.
si	Set-Item	Changes the value of an item to the value

9. Alias

These are shortcut commands that map to longer ones which means less typing.

%	ForEach-Object
?	Where-Object
ac	Add-Content
asnp	Add-PSSnapin
cat	Get-Content
cd	Set-Location
CFS	ConvertFrom-String
chdir	Set-Location
clc	Clear-Content
clear	Clear-Host
clhy	Clear-History
cli	Clear-Item
clp	Clear-ItemProperty
cls	Clear-Host
clv	Clear-Variable
cnsn	Connect-PSSession
compa re	Compare-Object
copy	Copy-Item
cp	Copy-Item
cpi	Copy-Item
cpp	Copy-ItemProperty
curl	Invoke-WebRequest
cvpa	Convert-Path
dbp	Disable-PSBreakpoint
del	Remove-Item
diff	Compare-Object
dir	Get-ChildItem
dnsn	Disconnect-PSSession
ebp	Enable-PSBreakpoint
echo	Write-Output
epal	Export-Alias
epcsv	Export-Csv
epsn	Export-PSSession
erase	Remove-Item
etsn	Enter-PSSession
exsn	Exit-PSSession
fc	Format-Custom
fhx	Format-Hex
fl	Format-List
foreac h	ForEach-Object

ft	Format-Table
fw	Format-Wide
gal	Get-Alias
gbp	Get-PSBreakpoint
gc	Get-Content
gcai	Get-CimAssociatedInstance
gcb	Get-Clipboard
gci	Get-ChildItem
gcim	Get-CimInstance
gcls	Get-CimClass
gcm	Get-Command
gcms	Get-CimSession
gcs	Get-PSCallStack
gdr	Get-PSDrive
ghy	Get-History
gi	Get-Item
gin	Get-ComputerInfo
gjb	Get-Job
gl	Get-Location
gm	Get-Member
gmo	Get-Module
gp	Get-ItemProperty
gps	Get-Process
gpv	Get-ItemPropertyValue
group	Group-Object
gsn	Get-PSSession
gsnp	Get-PSSnapin
gsv	Get-Service
gtz	Get-TimeZone
gu	Get-Unique
gv	Get-Variable
gwmi	Get-WmiObject
h	Get-History
history	Get-History
icim	Invoke-CimMethod
icm	Invoke-Command
iex	Invoke-Expression
ihy	Invoke-History
ii	Invoke-Item
ipal	Import-Alias

ipcsv	Import-Csv
ipmo	Import-Module
ipsn	Import-PSSession
irm	Invoke-RestMethod
ise	powershell_ise.exe
iwmi	Invoke-WmiMethod
iwr	Invoke-WebRequest
kill	Stop-Process
lp	Out-Printer
ls	Get-ChildItem
man	help
md	mkdir
measure	Measure-Object
mi	Move-Item
mount	New-PSDrive
move	Move-Item
mp	Move-ItemProperty
mv	Move-Item
nal	New-Alias
ncim	New-CimInstance
ncms	New-CimSession
ncso	New-CimSessionOption
ndr	New-PSDrive
ni	New-Item
nmo	New-Module
npssc	New-PSSessionConfigurationFile
nsn	New-PSSession
nv	New-Variable
ogv	Out-GridView
oh	Out-Host
popd	Pop-Location
ps	Get-Process
pushd	Push-Location
pwd	Get-Location
r	Invoke-History
rbp	Remove-PSBreakpoint
rcie	Register-CimIndicationEvent
rcim	Remove-CimInstance
rcjb	Receive-Job
rcms	Remove-CimSession
rcsn	Receive-PSSession

rd	Remove-Item
rdr	Remove-PSDrive
ren	Rename-Item
ri	Remove-Item
rjb	Remove-Job
rm	Remove-Item
rmdir	Remove-Item
rmo	Remove-Module
rni	Rename-Item
rnp	Rename-ItemProperty
rp	Remove-ItemProperty
rsn	Remove-PSSession
rsnp	Remove-PSSnapin
rujb	Resume-Job
rv	Remove-Variable
rvpa	Resolve-Path
rwmi	Remove-WmiObject
sajb	Start-Job
sal	Set-Alias
saps	Start-Process
sasv	Start-Service
sbp	Set-PSBreakpoint
sc	Set-Content
scb	Set-Clipboard
scim	Set-CimInstance
select	Select-Object
set	Set-Variable
shcm	Show-Command
si	Set-Item
sl	Set-Location
sleep	Start-Sleep
sls	Select-String
sort	Sort-Object
sp	Set-ItemProperty
spjb	Stop-Job
spps	Stop-Process
spsv	Stop-Service
start	Start-Process
stz	Set-TimeZone
sujb	Suspend-Job
sv	Set-Variable
swmi	Set-WmiInstance
tee	Tee-Object
trcm	Trace-Command
type	Get-Content

wget	Invoke-WebRequest
where	Where-Object

wjb	Wait-Job
write	Write-Output