# PowerShell Handbook

**Raman Deep Singh**

---

PowerShell is a shell for Windows 10 and Windows11.

For this tutorial Run Windows Power Shell as Administrator

command to find version of powershell

```
$Host
```

___________________________________________________________

command to find ip address of current system on powershell

```
ipconfig
```

____________________________

command to find all files and directories in the current directory

```
dir
```

_____________________________________

command to make a directory

```
md test
```

_________________________________

command to clear screen

```
clear
```

or

```
cls
```

_______________________________________

What are Cmdlet in Powershell

Cmdlet is a command that gives some output
example

```
Get-Service
```

Above cmdlet will give list of all services on the windows

_____________________________________________________________

cmdlet with parameters
here name is a parameter and every parameter begins with a hyphen(-)
here spooler is a value passed for the parameter

```
get-service -name spooler
```

will give details about spooler service

_______________________________________________

parameters with spaces requires quotes

```
get-service -DisplayName 'print spooler'
```

will give all services where displayname is print spooler
___________________________________________

how to stop a service

```
stop-service -name 'Print Spooler'
```

how to start a service

```
start-service -name 'Print Spooler'
```

---

cmdlet to get Local Users

```
Get-LocalUser
```

---

use cmdlet set-localuser to set description of the local user account

```
set-localuser -name 'raman' -description 'this is raman account'
```

---

how to get help for a cmdlet

```
Get-Help -Name Get-Service
```

---

how to disable a local user

```
Disable-LocalUser -Name 'raman'
```

how to enable a local user

```
Enable-LocalUser -Name 'raman'
```

---

all powershell scripts have extension (.ps1)

you can run .ps1 file using powershell.exe

create a c:\temp\abc.ps1 with following data

---

```
dir
```

---

start command prompt and type

```
powershell.exe c:\temp\abc.ps1
```

if script doesn't executes open powershell as administrator

and type following cmdlet

```
Set-ExecutionPolicy RemoteSigned
```

After running the above cmdlet you will be able to run .ps1 file as Administrator

---

Module in Powershell is a group of cmdlets

The Get-Command cmdlet gets all commands that are installed on the computer, including cmdlets, aliases, functions, filters, scripts, and applications.

---

Pipeline in PowerShell

| is ised for pipelines in Powershell

Following are example of Pipeline

```
Get-Service -Name 'spooler' | Stop-Service
```

```
Get-Service -Name 'spooler' | Start-Service
```

Another example of pipeline is as below

```
Get-LocalUser -Name 'raman' | Set-LocalUser -Description 'description of my account'
```

---

How to get details of a file using Get-Item Cmdlet

```
Get-Item -Path c:\temp\file1.txt
```

---

How to copy a file from source to destination using Copy-Item Cmdlet
the folder temp1 should exist

```
Copy-Item -Path 'c:\temp\file1.txt' -Destination 'c:\temp1'
```

---

How to copy a file from a folder to another folder using a pipeline
folder c:\temp1\files2 should exist

```
Get-Item -Path 'c:\temp\file1.txt' | Copy-Item -Destination 'c:\temp1\files2'
```

______________________________________________________________________

Get list of all files in c:\temp using Get-ChildItem

```
Get-ChildItem -Path 'c:\temp'
```

Copy all files in c:\temp to c:\temp2 using pipeline and Get-ChildItem cmdlet

folder c:\tenp2 should exist

```
Get-ChildItem -Path 'c:\temp' | Copy-Item -Destination 'c:\temp2'
```

______________________________________________________________________

How to output result of a cmdlet to a text file

```
Get-Service | Out-File -FilePath 'c:\temp\services.txt'
```

Above cmdlet will get all the list of services on the computer and will store this list in a text file c:\temp\services.txt

folder c:\temp should exist

This is an example of pipeline and Out-File is a Cmdlet

_____________________________________________________________________

Formatting Data in Powershell

```
Get-Service | Format-List
```

Above cmdlet will get list of all the services and will format the output as a list

```
Get-Service | Format-Table
```

Above cmdlet will get list of all the services and will format the output as a table

```
Get-Service | Format-Table -Autosize | Out-File -FilePath 'c:\temp\servicestable.txt'
```

Above cmdlet will get list of all services and will format the output as a table and will also store output in a file

c:\temp\servicestable.txt and folder c:\temp should exist

______________________________________________________________________

Getting all members of an object

```
Get-Service | Get-Member
```

Above cmdlet will get all properties and methods of object Service

```
Get-LocalUser | Get-Member
```

Above cmdlet will get all properties and methods of object LocalUser

____________________________________________________________________

Types of properties in powershell

string

bool

int

char

________________________________________________________________

Example of Where-Object which is used to filter properties of object

$PSItem is the value of the current object being validated

Following cmdlet will give list of all running services and will store output in c:\temp\report.txt

```
Get-Service | Where-Object {$PSItem.status -eq 'running'} | Out-File -FilePath 'c:\temp\report.txt'
```

________________________________________________________________

Following cmdlet will set description of all the diabled Local User Accounts to Disabled Account

```
 Get-LocalUser | Where-Object {$PSItem.enabled -eq $false} | Set-LocalUser -Description 'Disabled Account'
```

________________________________________________________________

Example of Select-Object to select specific information like getting first 5 rows of the output

```
Get-Service | Select-Object -First 5
```

Above cmdlet will print first five services from the list of services on the computer

```
Get-Service | Select-Object -Last 5
```

Above cmdlet will print last five services from the list of services on the computer

```
Get-Service | Select-Object -Property name,status -First 5 | Out-File -FilePath 'c:\temp\report3.txt'
```

Above cmdlet will print name and status property of first 5 services to a file c:\temp\report3.txt

______________________________________________________________________________

Sort-Object is used to sort a list based on property of object

```
Get-Service | Sort-Object -Property 'Status'
```

______________________________________________________________________________

Output results to csv file

```
Get-LocalUser | Export-Csv -Path 'c:\temp\report.csv'
```

Output results to csv file with no type information, no type information means csv file will not contain type of the object

```
Get-LocalUser | Export-Csv -Path 'c:\temp\report1.csv' -NoTypeInformation
```

Output results to a csv file by giving a delimiter as -, by default delimiter in csv file is ,

```
Get-LocalUser | Export-Csv -Path 'c:\temp\report1.csv' -NoTypeInformation -Delimiter '-'
```

______________________________________________________________________________

```
Get-Service |Select-Object -Property name,status | Export-Csv 'c:\temp\report45.csv'
```

Above cmdlet will get list of all services and will print two properties of service object name and status to a csv file

'c:\temp\report45.csv'

______________________________________________________________________________

create a new local user

```
New-LocalUser -Name 'ramandeep'-NoPassword
```

---

```
Get-LocalUser | Where-Object {$PSItem.Enabled -eq $true}
```

get all local users where Enabled is true or local user is enabled

---

#1

```
New-Item -Path 'C:\MyDir' -ItemType Directory
```

#2

```
New-Item -Path 'C:\MyDir\1.txt' -ItemType Directory
New-Item -Path 'C:\MyDir\2.txt' -ItemType Directory
```

#3

```
New-Item -Path 'C:\MyDir\1.txt' -ItemType Directory
New-Item -Path 'C:\MyDir\2.txt' -ItemType Directory
```

#4

```
Get-ChildItem -Path 'C:\MyDir' | Remove-Item -Force
```

#5

```
Get-ChildItem -Path 'C:\MyDir'| Where-Object {$PSItem.Extension -eq '.txt'} | Remove-Item -Force
```

#6

```
Get-ChildItem -Path 'C:\MyDir'| Where-Object {$PSItem.Extension -eq '.bmp'} | Move-Item -
Destination 'C:\moved'
```

#7

```
New-LocalUser -Name 'user1' -Description 'sales' -NoPassword

New-LocalUser -Name 'user2' -Description 'sales' -NoPassword

New-LocalUser -Name 'user3' -Description 'support' -NoPassword

New-LocalUser -Name 'user4' -Description 'support' -NoPassword
```

#8

```
Get-LocalUser | Where-Object {$PSItem.Description -eq 'sales'} | Disable-LocalUser

Get-LocalUser | Where-Object {$PSItem.Description -eq 'sales'} | Set-LocalUser -Description
'disabled'
```

#9

```
Get-Process | Where-Object {$PSItem.HandleCount -lt 100}
```

______________________________________________________________________

What is Powershell ISE?

Powershell ISE is Integrated Scripting Environment.

For following code we will be using Powershell ISE

Start Powershell ISE by typing Powershell ISE in search bar.

to run your scripts ISE run command in powershell terminal

```
Set-ExecutionPolicy Unrestricted
```

or

```
Set-ExecutionPolicy RemoteSigned
```

______________________________________________________________________

script code to print Hello World in ISE

save the following code in hello.ps1

and you can also run script as .\hello.ps1

which will give output as Hello World

```
'Hello World'
```

________________________________________________________

Following is an example of variable

In the following code we create a variable Name with value as Raman Deep Singh and then we have printed value in variable Name

```
$Name='Raman Deep Singh'
$Name
```

---

example of concatenation in powershell

```
$Name='Raman Deep Singh'
'Hello ' + $Name
```

---

Different types of variables

Integer (int)

String (str or text)

Boolean (bool)

DateTime (to store date)

defining datatypes

following example print message Hello Raman Deep Singh

it also gives variable $Name datatype as string

it gives variables $a,$b,$c datatype as int

it prints sum of two number values in variable a and b

```
[string]$Name='Raman Deep Singh'
'Hello ' + $Name

[int]$a=10
[int]$b=20
```

```
[int]$c=$a+$b
 'After addition of a and b value of c is ' + $c
```

Output

```
Hello Raman Deep Singh
After addition of a and b value of c is 30
```

___

How to get Datatypes of variables

```
[string]$Name='Raman Deep Singh'
'Hello ' + $Name

[int]$a=10

$Name.GetType()

$a.GetType()
```

___

Output

```
IsPublic IsSerial Name                                     BaseType
-------- -------- ----                                     --------
True     True     String                                   System.Object
True     True     Int32                                    System.ValueType
```

___

Example of Read-Host

Read-Host is sued to input data from User

Example is as below

___________________________________________

```
'Enter Your First Name '

$FirstName=Read-Host


'Enter Your Last Name'

$LastName=Read-Host


'Your Name is ' + $FirstName + ' ' + $LastName
```

Output

```
Enter Your First Name

Raman Deep

Enter Your Last Name

Singh

Your Name is Raman Deep Singh
```

________________________________________________

Program to find sum of two numbers

```
'Enter First Number'

[int]$a=Read-Host

'Enter Second Number'

[int]$b=Read-Host

[int]$c=$a+$b

'Sum of two numbers is ' + $c
```

Output

```
Enter First Number
10
Enter Second Number
20
Sum of two numbers is 30
```

---

Same program can be modified for operators

- (subtraction)

* (multiplication)

/ (division)

% or Mod (remainder)

---

Example of storing an object in a variable

```
$Service=Get-Service -Name 'spooler'
$Service | Get-Member
```

Output will be list of all members of spooler Service Object

---

Example of is if-elseif statement

```
$Day = 'Wednesday'

if($Day -eq 'Monday')
{
    'On Monday you eat Bami Goreng'

}

elseif($Day -eq 'Tuesday')
{

    'On Tuesday you eat Rendang'
    'Cause you like that '

}

elseif($Day -eq 'Wednesday')
{

    'On Wednesday you eat Nasi Goreng'

}

else
```

```
{

    'I Do not care !!! '

}
```

Output

```
On Wednesday you eat Nasi Goreng
```

________________________________________________________________

Example to get a username from user and if LocalUser with this name is enabled set it to disable if it is disabled set it to enabled

Example of storing a User Object in a variable and also example of if elseif statement

```
'Hello give me a user ! '

$UserName = Read-Host

$UserObject = Get-LocalUser -Name $UserName

$UserObject


if($UserObject.Enabled -eq $true)
{
    Disable-LocalUser -Name $UserName


}

elseif($UserObject.Enabled -eq $false)
{


    Enable-LocalUser -Name $UserName


}


Get-LocalUser -Name $UserName
```
______________________________________________________________________________

Example of Comparison Operators

```
'Give me number 1'
[int]$Number1 = Read-Host

'Give me number 2'
[int]$Number2 = Read-Host



if($Number1 -gt $Number2)
{
    'Number1 is bigger than Number2'

}

if($Number1 -lt $Number2)
{

    'Number1 is less than Number2'

}

else
{

    'Number1 is equal to number2'

}
```

_____________________________________________________________________

Example of Logical Operators

```
'Give me your name '

$Name = Read-Host
```

```
'Give me your password'


$PassWord = Read-Host

if($Name -eq 'David'-and $PassWord -eq 'cat123') #If you want to match case use -ceq (So
PowerShell looks at lower and upper case

{



    'Welcome '+$Name



}


elseif($Name -eq 'Jan'-and $PassWord -eq 'fish123') #If you want to match case use -ceq (So
PowerShell looks at lower and upper case

{



    'Welcome '+$Name



}

else
```

```
{

    'Name or password not correct'

}
```

Output

```
Give me your name
David
Give me your password
cat123
Welcome David
```

______________________________________________________________________________

Example of while loop

```
$Day = 'Monday'

while($Day -eq 'Monday')
{

    'It is '+ $Day  + ' what day is it now ? '

    $Day = Read-Host

    sleep -Seconds 1
}

'Ok now it is not Monday any more !! it is '+$Day
```

Output

```
It is Monday what day is it now ?
Monday
It is Monday what day is it now ?
Tuesday
Ok now it is not Monday any more !! it is Tuesday
```

______________________________________________________________

Another example of while loop

```
[int]$Number1 = 1   #Make sure we have an integer with [int]$Number1 because Read-Host provides a string be default and we want an integer.We only need to do that once

[int]$Number2 = 2   #Make sure we have an integer with [int]$Number2 because Read-Host provides a string be default and we want an integer.We only need to do that once


while($Number1 -le $NUmber2)
{



   'Number1 is less than number2'


   'Provide number 1'
    $Number1 = Read-Host


   'Provide number 2'
```

```
    $Number2 = Read-Host

}

'End script! '
```

Output

```
Number1 is less than number2
Provide number 1
1
Provide number 2
2
Number1 is less than number2
Provide number 1
3
Provide number 2
2
End script!
```

---

Example of Array in Powershell

```
$Array = @()

$Array = $Array + 'Pete'

$Array = $Array + 'Linda'
```

```
$Array = $Array + 'John'

'You are nice '+ $Array[0]

'You are nice '+ $Array[1]

'You are nice '+ $Array[2]
```

Output

```
You are nice Pete
You are nice Linda
You are nice John
```

---

Example of traversing an array using foreach statement

```
$Array = @()

$Array = $Array + 'Pete'

$Array = $Array + 'Linda'

$Array = $Array + 'John'


foreach($Item in $Array)
{


   'That is a nice name '+ $Item
}
```

Output

```
You are nice Pete
You are nice Linda
You are nice John
```

________________________________________________________________________________

Example of storing all the Local users in an array and printing the description

```
$LocalUsers = Get-LocalUser



'The '+ $LocalUsers[0].Name + ' has a description '+ $LocalUsers[0].Description

'The '+ $LocalUsers[1].Name + ' has a description '+ $LocalUsers[1].Description

'The '+ $LocalUsers[8].Name + ' has a description '+ $LocalUsers[8].Description
```

____________________________________________________________________________________

Example of storing all Local Users Objects in array and traversing the array using foreach statement

```
$LocalUsers = Get-LocalUser
```

```
foreach($Item in $LocalUsers)

{




    'The '+ $Item.Name + ' has a description '+ $Item.Description




}
```

________________________________________________________________________________