Powershell Tutorial for Beginners: Learn in 1 Day

What is PowerShell?

Windows PowerShell is object-oriented automation engine and scripting language. It is designed mainly for the system administrators. It helps IT, professionals, to control & automate the administration of the Window OS and other applications.

It introduced some compelling new concepts that enable you to extend the

the knowledge you have gained and the scripts that you have created within the Windows Command Prompt and Windows Script Host environments.

It combines the flexibility of scripting, command-line speed, and the power of a GUI-based admin tool. It allows you to solve problems efficiently by helping system admin to eliminate future manual labor hours. We will go through all the important aspect which you should know to learn PowerShell.

In this training course, you will learn

This is a complete guide to PowerShell… let's begin!

Why Use Powershell?

Here, are some important reason for using Powershell:

- Powershell offers a well-integrated command-line experience for the operation system
- PowerShell allows complete access to all of the types in the .NET framework
- Trusted by system administrators.
- PowerShell is a simple way to manipulate server and workstation components
- It's geared toward system administrators by creating a more easy syntax
- PowerShell is more secure than running VBScript or other scripting languages

Features of Powershell

- **PowerShell Remoting**: PowerShell allows scripts and cmdlets to be invoked on a remote machine.
- **Background Jobs**: It helps you to invoked script or pipeline asynchronously. You can run your jobs either on the local machine or multiple remotely operated machines.
- **Transactions**: Enable cmdlet and allows developers to perform
- **Evening:** This command helps you to listen, forwarding, and acting on management and system events.
- **Network File Transfer:** Powershell offers native support for prioritized, asynchronous, throttled, transfer of files between machines using the Background Intelligent Transfer Service (BITS) technology.

PowerShell Cmdlet

A cmdlet which is also called Command let is a is a lightweight command used in the Window base PowerShell environment. PowerShell invokes these cmdlets in the command prompt. You can create and invoke cmdlets command using PowerShell APIS.

Cmdlet vs. Command:

Cmdlets are different from commands in other command-shell environments in the following manners −

- Cmdlets are .NET Framework class objects It can't be executed separately
- Cmdlets can construct from as few as a dozen lines of code
- Parsing, output formatting, and error presentation are not handled by cmdlets
- Cmdlets process works on objects. So text stream and objects can't be passed as output for pipelining
- Cmdlets are record-based as so it processes a single object at a time

Most of the PowerShell functionality comes from Cmdlet's which is always in verb-noun format and not plural. Moreover, Cmdlet's return objects not text. A cmdlet is a series of commands, which is more than one line, stored in a text file with a .ps1 extension.

A cmdlet always consists of a verb and a noun, separated with a hyphen. Some of the verbs use for you to learn PowerShell is:

- **Get** — To get something
- **Start** — To run something
- **Out** — To output something
- **Stop** — To stop something that is running
- **Set** — To define something
- **New** — To create something

PowerShell commands

Following is a list of important PowerShell Commands:

Get-Help: Help about PowerShell commands and topics

Example: Display help information about the command Format-Table

```
Get-Help Format-Table
```

```
PS C:\Users\Admin> Get-Help Format-Table

NAME
    Format-Table

SYNTAX
    Format-Table [[-Property] <Object[]>] [-AutoSize] [-HideTable
    [<CommonParameters>]


ALIASES
    ft


REMARKS
    Get-Help cannot find the Help files for this cmdlet on this c
        -- To download and install Help files for the module that
        -- To view the Help topic for this cmdlet online, type: "
           go to https://go.microsoft.com/fwlink/?LinkID=113303.
```

Get-Command: Get information about anything that can be invoked

Example: To generate a list of cmdlets, functions installed in your machine

```
Get-Command
```

Windows PowerShell

```
PS C:\Users\Admin> Get-Command

CommandType     Name                                               Version    Source
-----------     ----                                               -------    ------
Alias           Add-ProvisionedAppxPackage                         3.0        Dism
Alias           Add-ProvisioningPackage                            3.0        Provisioning
Alias           Add-TrustedProvisioningCertificate                 3.0        Provisioning
Alias           Apply-WindowsUnattend                              3.0        Dism
Alias           Disable-PhysicalDiskIndication                     2.0.0.0    Storage
Alias           Disable-StorageDiagnosticLog                       2.0.0.0    Storage
Alias           Enable-PhysicalDiskIndication                      2.0.0.0    Storage
Alias           Enable-StorageDiagnosticLog                        2.0.0.0    Storage
Alias           Flush-Volume                                       2.0.0.0    Storage
Alias           Get-DiskSNV                                        2.0.0.0    Storage
Alias           Get-PhysicalDiskSNV                                2.0.0.0    Storage
Alias           Get-ProvisionedAppxPackage                         3.0        Dism
Alias           Get-StorageEnclosureSNV                            2.0.0.0    Storage
Alias           Initialize-Volume                                  2.0.0.0    Storage
Alias           Move-SmbClient                                     2.0.0.0    SmbWitness
Alias           Optimize-ProvisionedAppxPackages                   3.0        Dism
Alias           Remove-EtwTraceSession                             1.0.0.0    EventTracingMana
Alias           Remove-ProvisionedAppxPackage                      3.0        Dism
Alias           Remove-ProvisioningPackage                         3.0        Provisioning
Alias           Remove-TrustedProvisioningCertificate              3.0        Provisioning
Alias           Set-EtwTraceSession                                1.0.0.0    EventTracingMana
Alias           Write-FileSystemCache                              2.0.0.0    Storage
```

Get-Service: Finds all cmdlets with the word 'service' in it.

Example: Get all services that begin with "vm"

```
Get-Service "vm*"
```

```
PS C:\Users\Admin> Get-Service "vm*"

Status   Name               DisplayName
------   ----               -----------
Stopped  vmicguestinterface Hyper-V Guest Service Interface
Stopped  vmicheartbeat      Hyper-V Heartbeat Service
Stopped  vmickvpexchange    Hyper-V Data Exchange Service
Stopped  vmicrdv            Hyper-V Remote Desktop Virtualizati...
Stopped  vmicshutdown       Hyper-V Guest Shutdown Service
Stopped  vmictimesync       Hyper-V Time Synchronization Service
Stopped  vmicvmsession      Hyper-V PowerShell Direct Service
Stopped  vmicvss            Hyper-V Volume Shadow Copy Requestor


PS C:\Users\Admin>
```

Get- Member: Show what can be done with an object

Example: Get members of the vm processes.

```
Get-Service "vm*" | Get-Member
```

```
PS C:\Users\Admin> Get-Service "vm*" | Get-Member


   TypeName: System.ServiceProcess.ServiceController

Name                      MemberType    Definition
----                      ----------    ----------
Name                      AliasProperty Name = ServiceName
RequiredServices          AliasProperty RequiredServices = ServicesDependedOn
Disposed                  Event         System.EventHandler Disposed(System.Object, System.EventArgs)
Close                     Method        void Close()
Continue                  Method        void Continue()
CreateObjRef              Method        System.Runtime.Remoting.ObjRef CreateObjRef(type requestedType)
Dispose                   Method        void Dispose(), void IDisposable.Dispose()
Equals                    Method        bool Equals(System.Object obj)
ExecuteCommand            Method        void ExecuteCommand(int command)
GetHashCode               Method        int GetHashCode()
GetLifetimeService        Method        System.Object GetLifetimeService()
GetType                   Method        type GetType()
InitializeLifetimeService Method        System.Object InitializeLifetimeService()
Pause                     Method        void Pause()
Refresh                   Method        void Refresh()
Start                     Method        void Start(), void Start(string[] args)
Stop                      Method        void Stop()
WaitForStatus             Method        void WaitForStatus(System.ServiceProcess.ServiceControllerStatus
CanPauseAndContinue       Property      bool CanPauseAndContinue {get;}
CanShutdown               Property      bool CanShutdown {get;}
CanStop                   Property      bool CanStop {get;}
Container                 Property      System.ComponentModel.IContainer Container {get;}
```

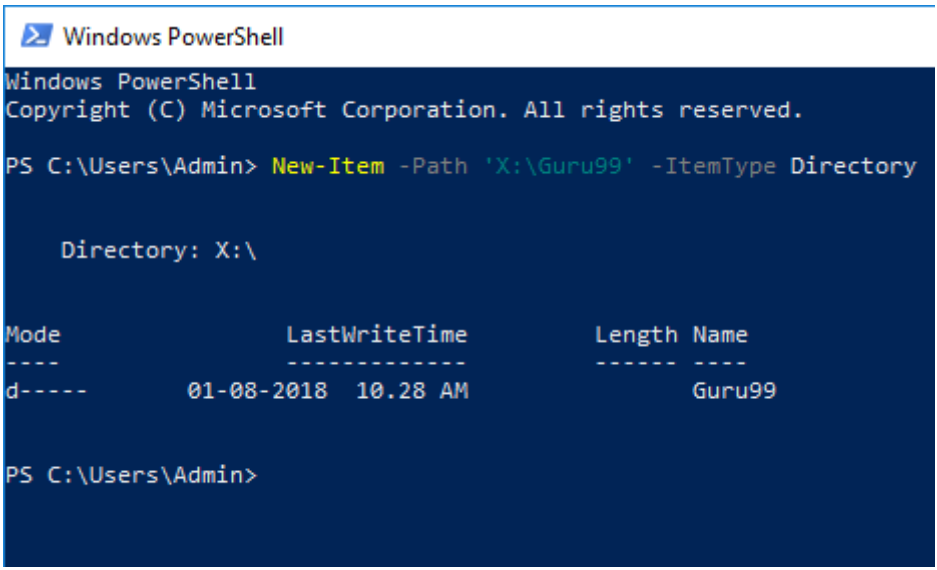
Other Commands:

- Get Module Shows packages of commands
- Get Content This cmdlet can take a file and process its contents and do something with it
- Get- get Finds all cmdlets starting with the word 'get-

Example: Create a Folder

```
New-Item -Path 'X:\Guru99' -ItemType Director
y
```

Output



Powershell Data types:

Data Type	Description
Boolean	True or false condition
Byte	An 8-bit unsigned whole number from to 255
Char	A 16-bit unsigned number from 0 to 63,535. For example, 1,026
Date	A calendar date, such as Auguest 3,2018
Decimal	A 128-bit decimal value, such as 5.18129265
Double	A double-precision 64-bit floating point number. This is another type of decimal value but has a very narrower range of values compared with a decimal data type.
Integer	A 32-bit signed whole number from -2,147,483.648 to 2,147.483.647. such as 15 or - 1932.
Long	A 64-bit signed whole number. This is like an integer but holds far bigger value. 9,238,372,039,854,775.877.
Object	Description
Short	A 16-bit unsigned number. This data type is similar to integer but holds far fewer values. It can only store values from -32,768 to 32,767.
Single	A single-precision 32-bit floating point number. This is a very similar data type just like double. However, it holds fewer values, such as 20.3654.
String	A grouping of characters which is also just called text

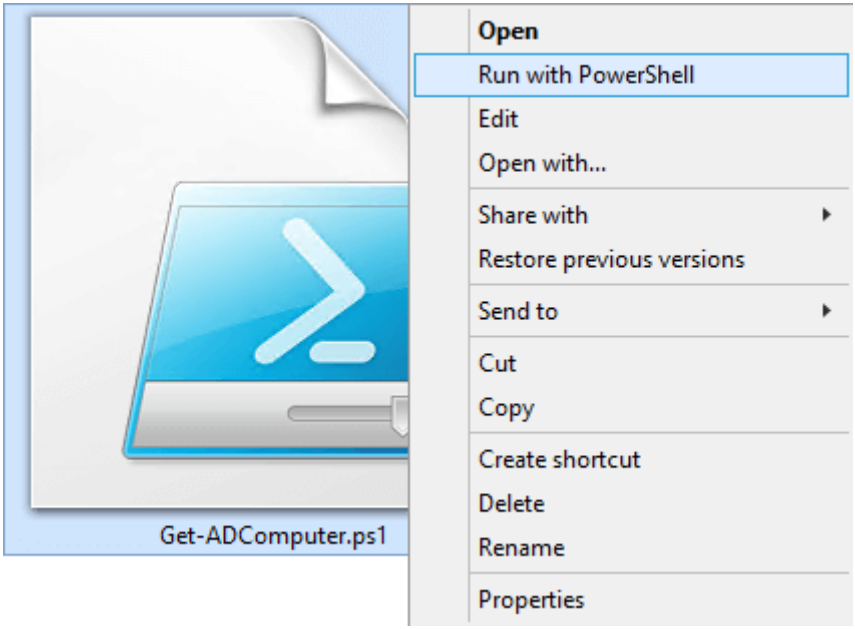
Special Variables

Special Variable	Description
$Error	An array of error objects which display the most recent errors
$Host	Display the name of the current hosting application
$Profile	Stores entire path of a user profile for the default shell
$PID	Stores the process identifier
$PSUICulture	It holds the name of the current UI culture.
$NULL	Contains empty or NULL value.
$False	Contains FALSE value
$True	Contains TRUE value

PowerShell Scripts

Powershell scripts are store in .ps1 file. By default, you can't run a script by just double-clicking a file. This protects your system from accidental harm. To execute a script:

Step 1: right-click it and click "Run with PowerShell."



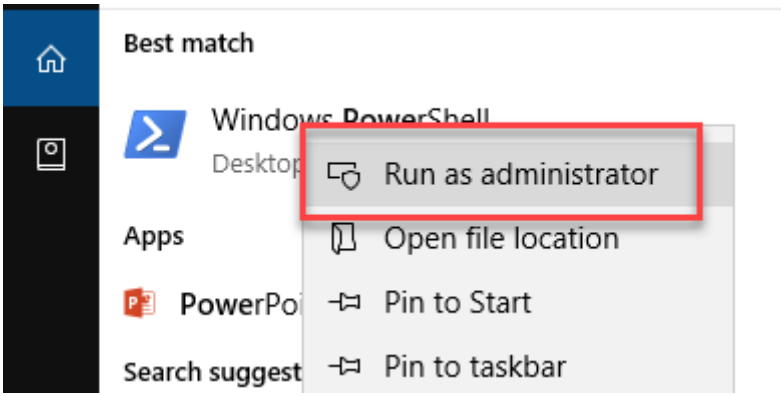
Moreover, there is a policy which restricts script execution. You can see this policy by running the Get-ExecutionPolicy command.

You will get one of the following output:

- **Restricted**— No scripts are allowed. This is the default setting, so it will display first time when you run the command.
- **AllSigned**— You can run scripts signed by a trusted developer. With the help of this setting, a script will ask for confirmation that you want to run it before executing.
- **RemoteSigned**— You can run your or scripts signed by a trusted developer.
- **Unrestricted**— You can run any script which you wants to run

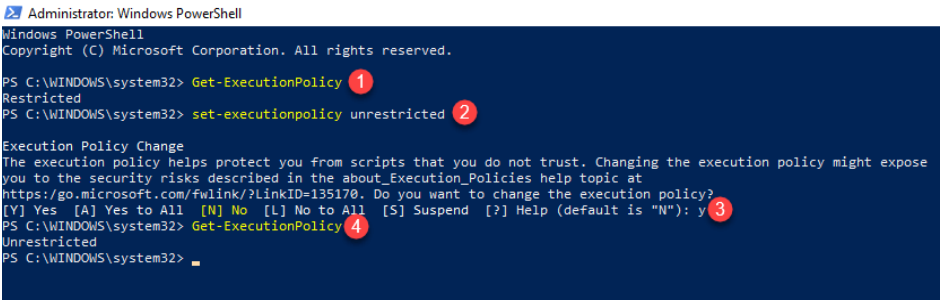
Steps to Change Execution Policy

Step 1) Open an elevated PowerShell prompt. Right Click on PowerShell and "Run as Administrator"



Step 2) Enter the Following commands

1. Get-ExecutionPolicy
2. Set-execution policy unrestricted
3. Enter Y in the prompt
4. Get-ExecutionPolicy

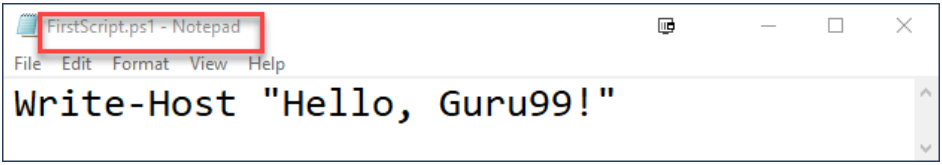


First PowerShell Script

In a notepad write the following command

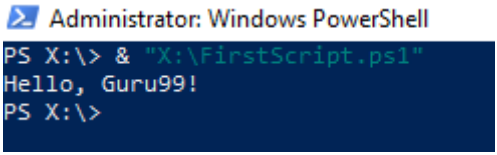
```
Write-Host "Hello, Guru99!"
```

PowerShell Scripts have an extension ps1. Save the file as FirstScript.ps1



In Powershell, call the script using the command

```
& "X:\FirstScript.ps1"
```

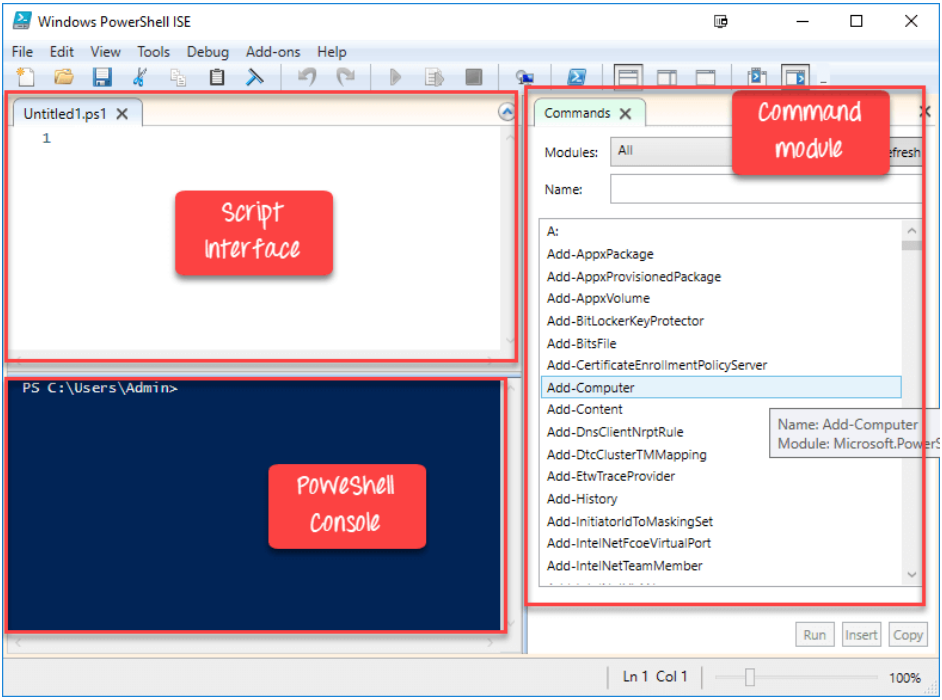


What is PowerShell ISE?

The Windows PowerShell Integrated Scripting Environment(ISE) is the default editor for Windows PowerShell. In this ISE, you can run commands, writer test, and debug scripts in an in a window base GUI environment. You can do multiline editing, syntax coloring, tab completion, selective execution and lots of other things.

Windows PowerShell ISE also allows you to run commands in a console pane. However, it also supports panes that you can use to simultaneously view the source code of your script and other tools which you can plug into the ISE.

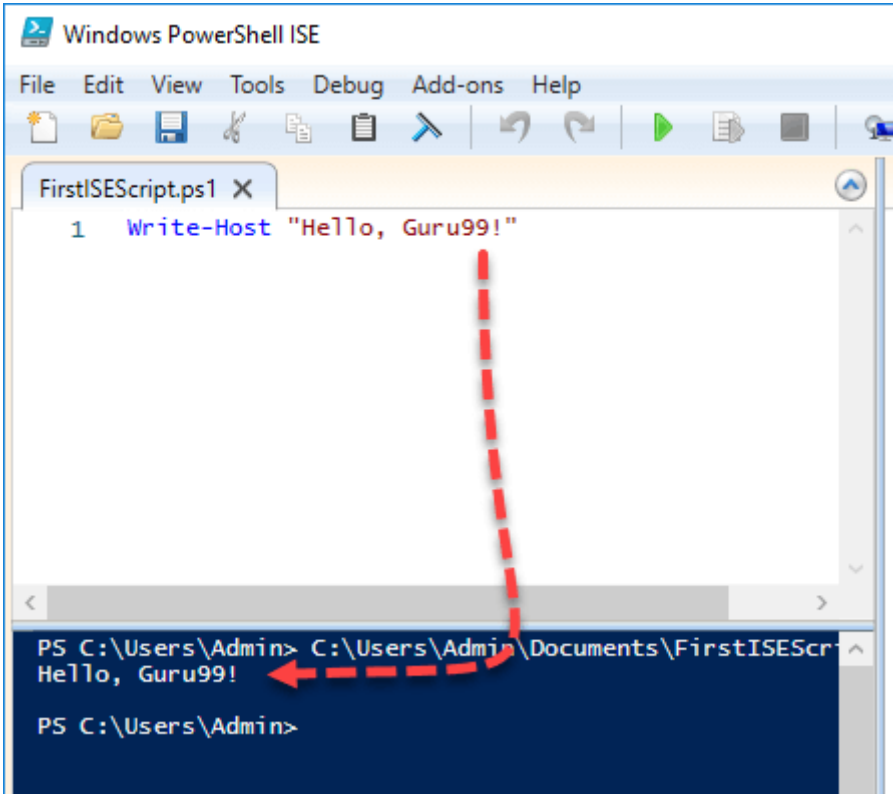
You can even open up multiple script windows at the same time. This is specifically useful when you are debugging a script which uses functions defined in other scripts or modules.



The same script we created in notepad, can be created in ISE

1. Paste code into the editor
2. Save Script

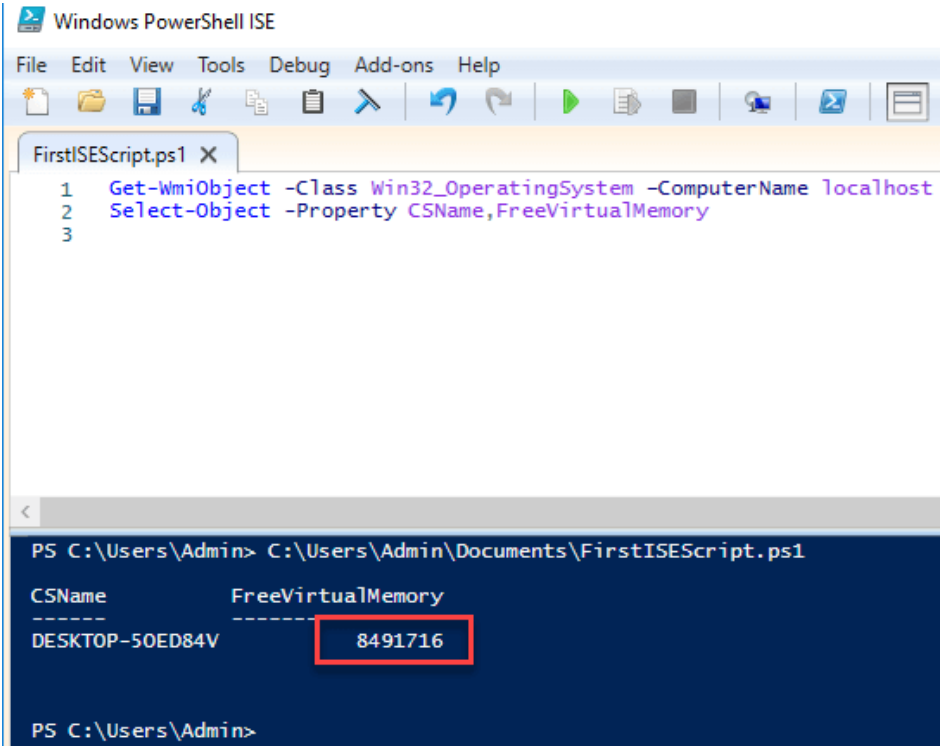
3. Use F5 to run the script
4. Observe output in the console



Sample 2:

The following code will give the Free Virtual Memory in your machine

```
Get-WmiObject -Class Win32_OperatingSystem –C
omputerName localhost |
Select-Object -Property CSName,FreeVirtualMem
ory
```



PowerShell Concepts

Cmdlets	Cmdlet are build-command written in .net languages like VB or C#. It allows developers to extend the set of cmdlets by loading and write PowerShell snap-ins.

Functions	Functions are commands which is written in the PowerShell language. It can be developed without using other IDE like Visual Studio and devs.
Scripts	Scripts are text files on disk with a .ps1 extension
Applications	Applications are existing windows programs.
What if	Tells the cmdlet not to execute, but to tell you what would happen if the cmdlet were to run.
Confirm	Instruct the cmdlet to prompt before executing the command.
Verbose	Gives a higher level of detail.
Debug	Instructs the cmdlet to provide debugging information.
ErrorAction	Instructs the cmdlet to perform a specific action when an error occurs. Allowed actions continue, stop, silently- continue and inquire.

ErrorVariable	It specifies the variable which holds error information.
OutVariable	Tells the cmdlet to use a specific variable to hold the output information
OutBuffer	Instructs the cmdlet to hold the specific number of objects before calling the next cmdlet in the pipeline.

Advantages of using PowerShell script

- PowerShell scripts are really powerful and could do much stuff in fewer lines.
- Variables are declared in the form $<variable>
- Variables could be used to hold the output of command, objects, and values.
- "Type" of a variable need not be specified.

PowerShell Vs. Command Prompt

PowerShell	Command Prompt

PowerShell deeply integrates with the Windows OS. It offers an interactive command line interface and scripting language.	Command Prompt is a default command line interface which provided by Microsoft. It is a simple win32 application that can interact and talk with any win32 objects in the Windows operating system.
PowerShell uses what are known as cmdlets. It can be invoked either in the runtime environment or the automation scripts.	No such features offer by command prompt.
PowerShell considers them as objects. So the output can be passed as an input to other cmdlets through the pipeline.	Command Prompt or even the *nix shell, the output generated from a cmdlet is not just a stream of text but a collection of objects.
The PowerShell is very advanced regarding features, capabilities and inner functioning.	Command prompt is very basic.

Applications of Powershell

Today, PowerShell has become an ideal choice for IT administrators as it eases management operation and effort in large corporate networks. For example, let's assume that you are managing a large network which contains more than four hundred servers. Now you want to implement a new security solution. This security solution depends on a certain service which needs to run on those servers.

You can surely log in to each server and see if they have that service install and running or not. However, it certainly takes a lot of human errors as your staff needs to spend lots of time on this non-productive process.

However, if you use PowerShell, then you could complete this task in just a few minutes. That's because the entire operation is done with a single script which gathers information about the services running on the servers.

Summary

- Windows PowerShell is object-oriented automation engine and scripting language
- Powershell offers a well-integrated command-line experience for the operation system
- PowerShell first version 1.0 was released in 2006
- PowerShell allows scripts and cmdlets to be invoked on a remote machine
- PowerShell is pre-installed in all latest versions of Windows
- A cmdlet is a lightweight command used in the Window base PowerShell environment
- Get, Start, Out, Stop, Set, New are important PowerShell commands

- Boolean, Byte, Chat, Decimal, Decimal, Long are important Data Type of PowerShell
- $Error. $Host, $Profile, $PID, $PSUICulture, $NULL are some special variable used in PowerShell
- The Windows PowerShell Integrated Scripting Environment(ISE) is the default editor for PowerShell
- PowerShell deeply integrates with the Windows OS whereas Command Prompt is a default command line interface which provided by Microsoft
- PowerShell has become an ideal choice for IT administrators as it eases management operation and effort in large corporate networks