**Powershell Tutorial for Beginners: Learn Powershell Scripting**

Let's start learning Powershell Lessons. First, we will learn:

**What is PowerShell?**

**Windows PowerShell** is an object-oriented automation engine and scripting language. It is designed mainly for IT professionals and system administrators to control & automate the administration of Windows OS and other applications. It provides compelling new concepts to extend the knowledge you have gained and scripts 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 aspects which you should know to learn PowerShell.

In this Powershell tutorial for beginners, you will learn Powershell basics like:

* [What is PowerShell?](https://www.guru99.com/powershell-tutorial.html#1)
* [Why Use Powershell?](https://www.guru99.com/powershell-tutorial.html#2)
* [PowerShell History](https://www.guru99.com/powershell-tutorial.html#3)
* [Features of Powershell](https://www.guru99.com/powershell-tutorial.html#4)
* [How to launch PowerShell](https://www.guru99.com/powershell-tutorial.html#5)
* [PowerShell Cmdlet](https://www.guru99.com/powershell-tutorial.html#6)
* [Cmdlet vs Command:](https://www.guru99.com/powershell-tutorial.html#7)
* [Powershell Data types:](https://www.guru99.com/powershell-tutorial.html#8)
* [Special Variables](https://www.guru99.com/powershell-tutorial.html#9)
* [PowerShell Scripts](https://www.guru99.com/powershell-tutorial.html#10)
* [First PowerShell Script](https://www.guru99.com/powershell-tutorial.html#11)
* [What is PowerShell ISE?](https://www.guru99.com/powershell-tutorial.html#12)
* [PowerShell Concepts](https://www.guru99.com/powershell-tutorial.html#13)
* [PowerShell Vs Command Prompt](https://www.guru99.com/powershell-tutorial.html#14)
* [Applications of Powershell](https://www.guru99.com/powershell-tutorial.html#15)

This is a complete guide to Powershell scripting basics… let's begin!

28.2M

277

How to write a TEST CASE Software Testing Tutorial

**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](https://www.guru99.com/vbscript-tutorials-for-beginners.html) or other scripting languages

**PowerShell History**

PowerShell first version 1.0 was released in 2006. Today, PowerShell is at version 5.1. As the year and version gone by, PowerShell's capabilities and hosting environments grew significantly.

**Let See Version wise History of Powershell:**

* PowerShell version 1 supported the local administration of Windows Server 2003
* PowerShell 2.0 was integrated with Windows 7 and Windows Server 2008 R2. This version supports for remoting and enhances the capabilities of PowerShell like transactions, background jobs, events, debugging, etc.
* PowerShell 3.0 was released as an internal part of the Windows management framework. It was installed on Windows 8 and Windows Server 2012. You can add and scheduled jobs, session connectivity, automatic module loading, etc.
* PowerShell 4.0 was shipped with Windows 8.1 and Windows Server 2012 R2. In this version added support for desired state configuration, enhanced debugging, network diagnostics.
* PowerShell 5.0 was released as internal part of Windows management framework 5. The feature offers in this version are remote debugging, class definitions, .NET enumerations, , etc.

Next in this Powershell scripting tutorial, we will learn about features of Powershell.

**Features of Powershell**

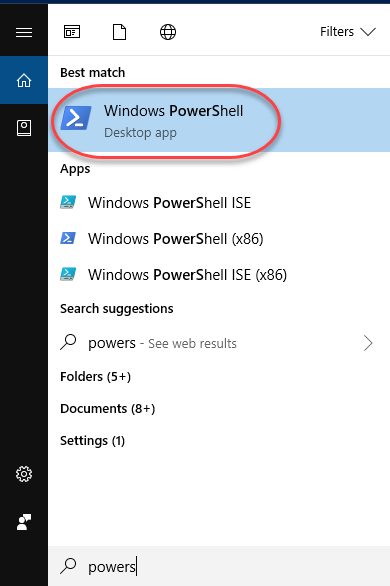
* **PowerShell Remoting**: PowerShell allows scripts and cmdlets to be invoked on a remote machine.
* **Background Jobs**: It helps you to invoked script or pipeline asynchronously. You can run your jobs either on the local machine or multiple remotely operated machines.
* **Transactions**: Enable cmdlet 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.

**How to launch PowerShell**

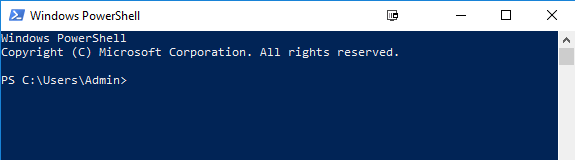
Now in this Powershell script tutorial, we will learn how to launch Powershell on Windows OS.

PowerShell is pre-installed in all latest versions of Windows. We need to launch PowerShell for that we need to follow the given steps:

**Step 1)** Search for PowerShell in Windows. Select and Click

[](https://www.guru99.com/images/tensorflow/082918_1415_PowershellT1.png)

**Step 2)** Power Shell Window Opens

[](https://www.guru99.com/images/tensorflow/082918_1415_PowershellT2.png)

**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](https://www.guru99.com/net-framework.html) 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

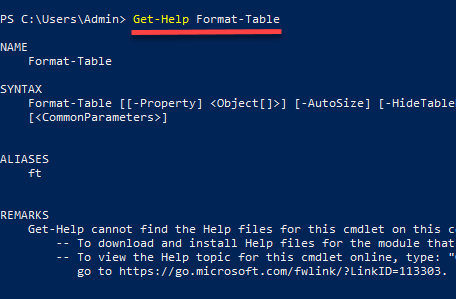
**PowerShell commands**

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

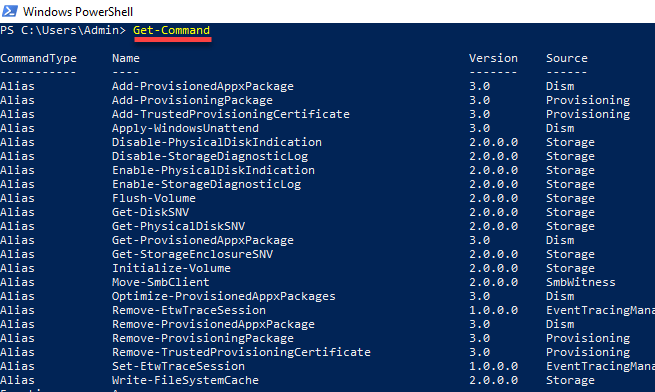
Get-Help Format-Table

[](https://www.guru99.com/images/tensorflow/082918_1415_PowershellT3.png)

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

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

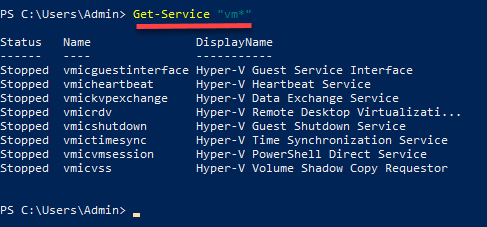
Get-Command

[](https://www.guru99.com/images/tensorflow/082918_1415_PowershellT4.png)

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

Example: Get all services that begin with "vm"

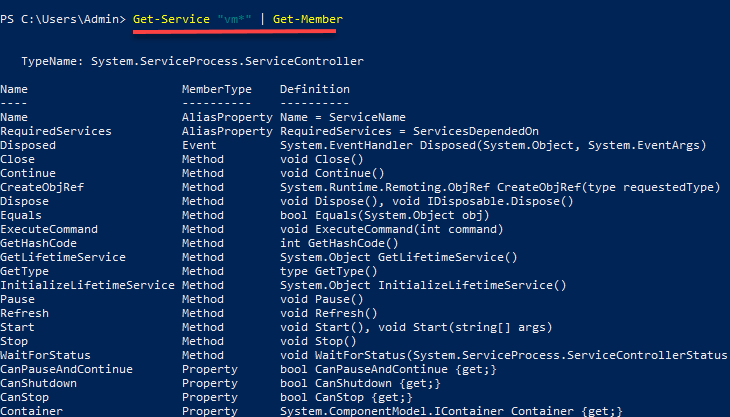
Get-Service "vm\*"

[](https://www.guru99.com/images/tensorflow/082918_1415_PowershellT5.png)

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

Example: Get members of the vm processes.

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

[](https://www.guru99.com/images/tensorflow/082918_1415_PowershellT6.png)

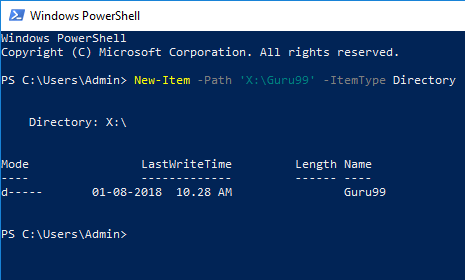
**Other Commands:**

* 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-

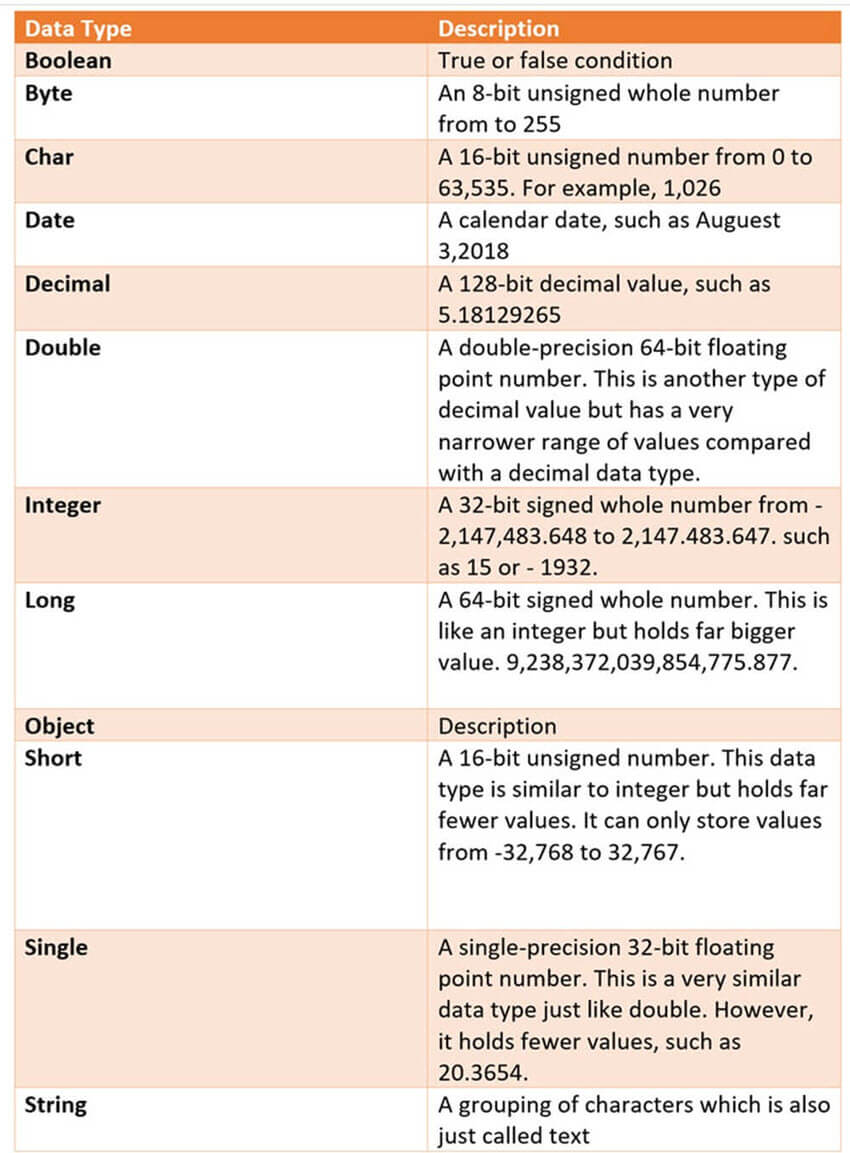
Example: Create a Folder

New-Item -Path 'X:\Guru99' -ItemType Directory

Output

[](https://www.guru99.com/images/tensorflow/082918_1415_PowershellT7.png)

**Powershell Data types:**

[](https://www.guru99.com/images/tensorflow/082918_1415_PowershellT8.jpg)

Powershell Data Types

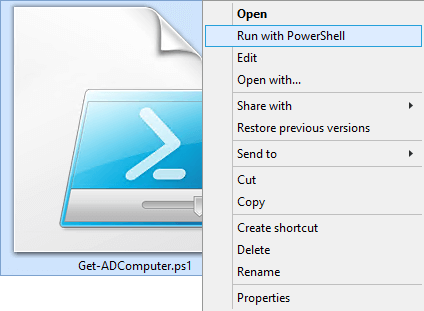
**Special Variables**

|  |  |
| --- | --- |
| **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."

[](https://www.guru99.com/images/tensorflow/082918_1415_PowershellT9.png)

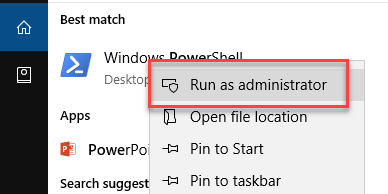
Moreover, there is a policy which restricts script execution. You can see this policy by running the Get-ExecutionPolicy command.

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

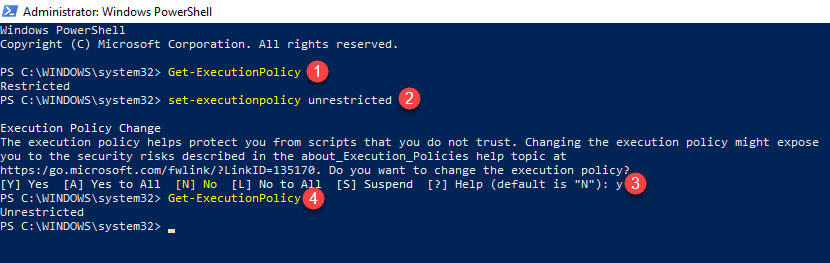
Steps to Change Execution Policy

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

[](https://www.guru99.com/images/tensorflow/082918_1415_PowershellT10.png)

**Step 2)** Enter the Following commands

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

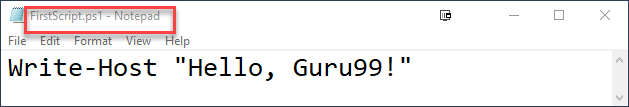
[](https://www.guru99.com/images/tensorflow/082918_1415_PowershellT11.png)

**First PowerShell Script**

In a notepad write the following command

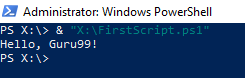
Write-Host "Hello, Guru99!"

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

[](https://www.guru99.com/images/tensorflow/082918_1415_PowershellT12.png)

In Powershell, call the script using the command

& "X:\FirstScript.ps1"

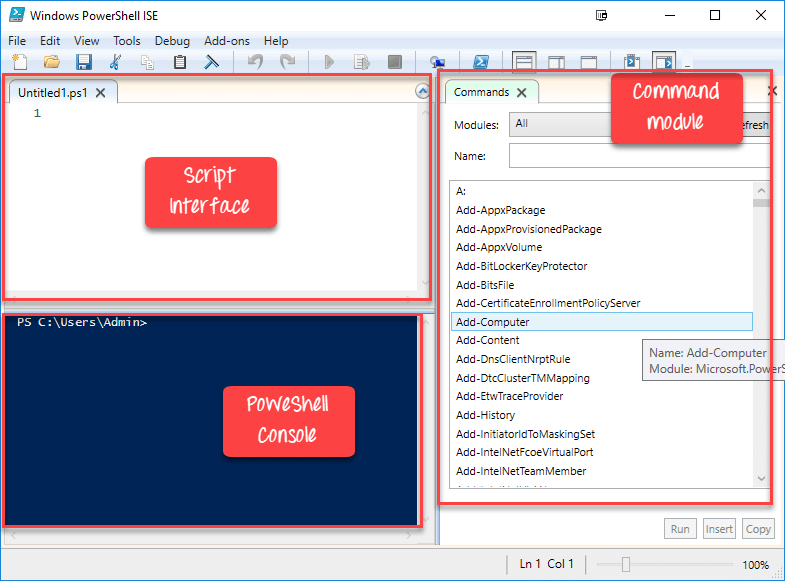
[](https://www.guru99.com/images/tensorflow/082918_1415_PowershellT13.png)

**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.

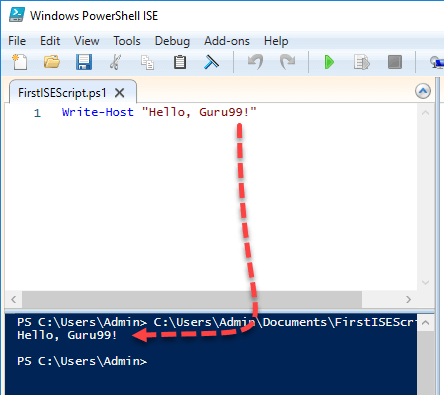
You can even open up multiple script windows at the same time. This is specifically useful when you are debugging a script which uses functions defined in other scripts or modules.

[](https://www.guru99.com/images/tensorflow/082918_1415_PowershellT14.png)

PowerShell ISE

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

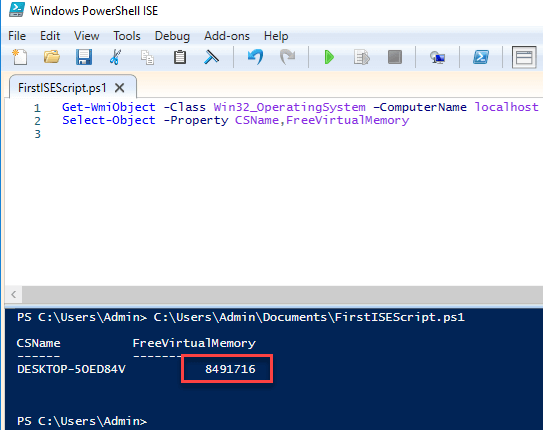
[](https://www.guru99.com/images/tensorflow/082918_1415_PowershellT15.png)

Sample 2:

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

Get-WmiObject -Class Win32\_OperatingSystem –ComputerName localhost |

Select-Object -Property CSName,FreeVirtualMemory

[](https://www.guru99.com/images/tensorflow/082918_1415_PowershellT16.png)

**PowerShell Concepts**

Now in this Powershell for beginners tutorial, we will learn about important 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