PowerShell Essential Guide

Master the fundamentals of PowerShell scripting and automation

Prashanth Jayaram

Rajendra Gupta

www.bpbonline.com

First published: 2024

Published by BPB Online
WeWork
119 Marylebone Road
London NW1 5PU

UK | UAE | INDIA | SINGAPORE

ISBN 978-93-55517-449

www.bpbonline.com

Dedicated to

Teachers, Family, Editors, and Friends

About the Authors

- **Prashanth Jayaram** brings 18 years of experience in Product Design, Data Engineering, Automation, and Data Science. He excels in crafting robust solutions and extracting insights from data across the fields of data engineering and science.

  He is passionate about sharing knowledge and have written extensively on a wide range of topics. He has authored and published over 500 articles, covering areas such as SQL, NoSQL, PowerShell, Python, SQL on Linux, SQL on Azure, SQL on AWS, MongoDB, Graph Database, Ansible, and Azure DevOps.

  The core of his professional journey revolves around the design and implementation of innovative database solutions. He had the privilege of immersing himself in cutting-edge Next-Gen database technologies, continually pushing the boundaries of what's possible in the database domain.

  His unwavering commitment to the SQL Server community has not gone unnoticed. He has been honored with prestigious awards, including the "ABOVE and BEYOND"recognition, underlining his exceptional contributions. In 2018, he was bestowed with the distinguished title of "Best SQL Author," a testament to his impact on the SQL Server world.

  His contributions extend beyond articles to the realm of book authorship. His books are comprehensive resources for those seeking knowledge and expertise in these domains.

- **Rajendra Gupta** is a distinguished Technical Architect and Database Consultant, boasting an impressive 17-year track record of immersive involvement in the realm of database technologies. With a remarkable accolade to his name, Rajendra has clinched the esteemed Best Author of the Year award from SQLShack not once, but thrice consecutively.

  An authoritative figure in the field, Rajendra has authored an extensive collection of over 600+ articles on SQLSHACK, MSSQLTIPS, QUEST, DZONE, and CodingSight that span across the expansive domains of SQL Server, Azure, MySQL, Linux, Power BI, Performance Tuning, AWS/Amazon RDS, Git, and various related technologies. His insightful compositions have garnered an impressive readership, with over 15 million avid readers engaging with his content to date.

  He has authored an extensive body of work, comprising more than 600 articles published on well-regarded platforms such as SQLSHACK, MSSQLTIPS, QUEST, DZONE, and CodingSight.

  These articles cover a wide array of subjects, including SQL Server, Azure, MySQL, Linux, Power BI, Performance Tuning, AWS/Amazon RDS, Git, and various other related technologies.

  He has consistently demonstrated his expertise by receiving the prestigious Best Author of the Year award from SQLShack for three consecutive years, reflecting his exceptional contributions to the industry.

  His comprehensive knowledge and insightful writings have not only benefited professionals in the field but have also earned a substantial and ever-growing readership. His contributions extend beyond articles to the realm of book authorship.

About the Reviewer

Shubham Chauhan is an accomplished IT professional holding a B.Tech degree in Electronics and Communication from the College of Technology, Pantnagar. His career in IT is characterized by a fervent commitment to automation, innovation, and excellence.

Shubham embarked on his IT journey at HCL Technologies, where he meticulously honed his skills in SCCM/Intune management and endpoint administration. Subsequently, at Stefanini India Pvt Ltd, he further refined his expertise, amassing invaluable experience in software deployment, patch management, and troubleshooting.

Throughout his illustrious career, Shubham has adeptly managed SCCM/Intune environments spanning 20,000 endpoints, setting himself apart with his exceptional proficiency in utilizing PowerShell to automate intricate tasks. This has led to substantial cost savings and operational improvements.

Currently serving as an Endpoint Administrator at Magnit, Shubham continues to harness the power of PowerShell and automation to optimize processes and enhance efficiency. He has consistently delivered cutting-edge solutions, optimizing software deployment, patch management, and troubleshooting.

Shubham's skill set extends to encompass software packaging, OS deployment, error resolution, and data analysis through SQL reporting. Collaborating seamlessly with cross-functional teams, he has successfully aligned technology solutions with organizational objectives, making valuable contributions to the company's success.

Acknowledgements

- **Prashanth Jayaram:** With heartfelt gratitude, I extend my appreciation to the remarkable individuals who have shaped my path with their kindness, humility, and empathy.

  To all the teachers, your passion for education, dedication to nurturing minds, and unwavering commitment to shaping our futures are deeply appreciated. You inspire us to learn, grow, and become better individuals. Thank you for your hard work, patience, and guidance.

  To my parents, R Jayaram and Indira, for the unwavering love and sacrifices that have been my cornerstone.

  To my beloved wife, Ambika S.G, your unwavering support and understanding have been my anchor. Your support knows no bounds.

  My twin angels, Prarthana and Pravitha, your joy and enthusiasm reflect the beauty of life itself. Your journey motivates me to aim higher. The sparkle in your mischievous eyes and your playful nature are a constant source of inspiration.

  To my siblings, Gopinath J and Balu J, for the great companionship that has been a steady anchor throughout my life.

  To my friend, Dr. Mallikarjuna A.M, your achievement, wisdom, and support have brightened my days and enriched my journey.

- **Rajendra Gupta:** Writing this book has been a journey of introspection, discovery, and dedication, and I am profoundly grateful for the many individuals who supported and inspired me along the way. Their contributions, encouragement, and presence have made this endeavor a reality.

  First and foremost, I extend my deepest gratitude to my family for their unwavering support. To my parents Mr. Shyam Sunder & Mrs. Krishna Gupta, who nurtured my love for learning and creativity from an early age, and to my kids —Akshita and Vanshil —whose belief in my abilities has been a constant motivation. Akshita, your wisdom and compassion inspire me daily, and Vanshil, your innocent curiosity brightens every moment.

  To my partner Kashish, whose patience, understanding, and belief in my dreams have been the cornerstone of this project. Your encouragement during moments of doubt and your celebration of every milestone have meant more to me than I can express.

  A heartfelt thank you to my friend Prashanth Jayaram, whose insightful discussions and unwavering support guided me through the creative process. Your friendship has been a source of inspiration and a reminder of the importance of collaboration.

Preface

The emphasis on automation is prevalent nowadays. Also, the importance of automation in today's world has driven us to propose this book.

In the last decade, PowerShell has propelled in every way in the automation arena. Since the inception of PowerShell, it has become a defacto tool for automation, and it is a favorite solution of many Windows administrators–with the capability to automate almost any task in the Microsoft ecosystem. Since the advent of PowerShell, it has been a lot easier to import the related modules and invoke the associated cmdlets call to take care of many day-to-day mundane activities, from simple to complex maintenance.

This book sets its course with insightful introductory chapters, encompassing a holistic view of PowerShell, complementary tools, and foundational concepts. As the narrative unfolds, it delves into advanced components, maintaining a balanced perspective that appreciates the IT administrator's stance while acknowledging the significance of automation GUI interfaces.

Our objective is to empower readers with practical, hands-on experience across diverse domains, including AD, Database, Cloud administration, Python, and integration tools.

The later chapters are straightforward to understand and completely isolated from each section. For every section, the PowerShell code is designed, and readers with no prior experience can jump into the topics and get started with the examples.

It would be great to have a hands-on experience with PowerShell; this would help you to progress faster. However, if you have experience with PowerShell, you can jump to a specific chapter or topic in the book. However, in some cases, if you are a beginner, you can start with the basics and build on that foundation.

Chapter 1: Introducing PowerShell - As is the tradition, we will take a few glimpses at the introduction to PowerShell, the history of PowerShell, look at what makes PowerShell so flexible and powerful, and introduce ourselves to the different landscapes of PowerShell integrations.

Chapter 2: PowerShell Constructs - In the second chapter, we will explore PowerShell constructs. PowerShell constructs are the building blocks of PowerShell scripts, including cmdlets, object-oriented notions, execution-policy, variables, pipelines, and modules.

Chapter 3: Munging the Data Through Pipelines - This chapter covers how to munge data through pipelines in PowerShell. Topics include selecting columns from the output, limiting the

number of output objects, expanding properties within properties, filtering objects, grouping the output, sorting the output, taking actions on returned objects, understanding pipeline-enabled parameters, and importing content into PowerShell. By the end of the chapter, you will have a deeper understanding of how to use PowerShell's cmdlets and features to work with large datasets more efficiently.

Chapter 4: Data Control Flow Using Branches and Loops - This chapter covers a wide range of PowerShell control flow and data manipulation concepts.

The control flow in PowerShell refers to the ability to define and control the data flow within a script. It involves executing or branching to commands based on the value of data, building logic that uses looping and branching, and creating conditions that determine which commands to execute based on certain conditions.

The If statement is one of PowerShell's most used branching constructs. Branching allows you to define multiple paths for your script based on certain conditions. It will enable you to test a condition and execute a block of code based on the test result.

Loops are another essential component of control flow in PowerShell. They allow you to execute a code block repeatedly while a particular condition is true. PowerShell supports several loop constructs, including For, ForEach, While, and Do-While.

Chapter 5: Learning about PowerShell Modules - In this chapter, you will learn more about PowerShell modules. You will gain a solid understanding of PowerShell modules and learn how to use them effectively.

You will learn how to create a module manifest, which provides metadata properties such as the module name, version, and author. You will also learn to specify common module metadata properties in a module manifest file. Furthermore, you will discover how to update an existing PowerShell module to a newer version. This chapter will cover best practices for preventing command name conflicts when working with multiple PowerShell modules.

Finally, you will understand how to utilize the PowerShell module path to locate and import modules into a PowerShell session. By the end of this chapter, you will be equipped with the knowledge and skills to use PowerShell modules effectively in your PowerShell projects.

Chapter 6: Choosing Between PowerShell Core and PowerShell - This chapter on PowerShell Core covers a range of subjects related to this cross-platform, open-source command-line shell and scripting language designed for automation and configuration management. It includes an introduction to PowerShell Core, a brief note on .NET Core, features of PowerShell Core, how to install it on Ubuntu, understanding the differences between PowerShell and PowerShell

Core, information about unsupported modules, learning about OpenSSH and remoting, and a comparison between the commands of Bash and PowerShell.

Chapter 7: PowerShell Administration and Scripting - In this chapter, you will learn how to deal with your daily work as an administrator. Also, you will learn how to build scripts to accomplish the automation for repetitive tasks and learn to generate various reports.

After the scripts are designed, written, deployed, and executed on a system, the scope can be local, remote, or background jobs. In addition, you will learn how to use the cmdlets to troubleshoot OS issues effectively.

Chapter 8: Using the Active Directory Module - Active Directory is the most widely used solution for administrating and managing users and resources within organizations.

Managing Active Directory (AD) with Windows PowerShell is more straightforward than many IT professionals think. PowerShell provides a powerful and intuitive management engine allowing for interactive AD management through its console.

With PowerShell, you can perform everyday AD management tasks without writing complex scripts. The interactive nature of PowerShell enables IT professionals to execute commands directly in the console, making it accessible to novice and experienced administrators. You can easily leverage PowerShell's command-line interface to perform various AD management operations.

In this chapter, you will learn more about active directory administration using PowerShell cmdlets. Also, you will see a list of the commonly used cmdlets to manage the AD, and we will discuss a list of typically performed operations. In addition, you will explore cmdlets to see what tasks you can automate.

Chapter 9: Building PowerShell GUI for Scripts - In this chapter you will learn, PowerShell is primarily designed to be used from the command line, it does have some graphical user interface (GUI) capabilities that can be leveraged using Windows Forms.

To create a GUI with PowerShell, you can use the Windows Forms classes that are part of the .NET Framework. Windows Forms allows you to create windows, controls, menus, and other graphical elements that can be used to create a user-friendly interface for your PowerShell scripts.

Chapter 10: Managing Cloud Operations Using PowerShell - This chapter explores the practical application of PowerShell in cloud management. Starting with an introduction to PowerShell and its relevance in cloud operations, the chapter emphasizes the widespread

adoption of PowerShell and its benefits. Readers are guided through preparing an Azure environment, loading Azure PowerShell modules, and establishing a connection to Azure. The chapter then covers essential tasks such as deploying Azure VMs, gathering details about VMs, and collecting metrics for monitoring and optimization. Furthermore, it introduces PowerShell modules for managing AWS and GCP resources, empowering readers with knowledge of multi-cloud management.

Chapter 11: Understanding PowerShell and Data Science - In this chapter you will learn, PowerShell is becoming increasingly associated with data science and machine learning due to its ability to work with data and integrate with data science tools and frameworks.

PowerShell includes several built-in features for working with data, such as support for regular expressions, XML and JSON parsing, and database connectivity.

Chapter 12: Administrating Database Using PowerShell - This chapter provides a comprehensive guide to administrating databases using PowerShell, covering various aspects such as understanding SQL Server, leveraging PowerShell modules, working with different data formats, and managing databases in both on-premises and Azure environments.

Code Bundle and Coloured Images

Please follow the link to download the
Code Bundle and the ***Coloured Images*** of the book:

https://rebrand.ly/yrb3s7h

The code bundle for the book is also hosted on GitHub at
https://github.com/bpbpublications/PowerShell-Essential-Guide.
In case there's an update to the code, it will be updated on the existing GitHub repository.

We have code bundles from our rich catalogue of books and videos available at
https://github.com/bpbpublications. Check them out!

Errata

We take immense pride in our work at BPB Publications and follow best practices to ensure the accuracy of our content to provide with an indulging reading experience to our subscribers. Our readers are our mirrors, and we use their inputs to reflect and improve upon human errors, if any, that may have occurred during the publishing processes involved. To let us maintain the quality and help us reach out to any readers who might be having difficulties due to any unforeseen errors, please write to us at :

errata@bpbonline.com

Your support, suggestions and feedbacks are highly appreciated by the BPB Publications' Family.

Piracy

If you come across any illegal copies of our works in any form on the internet, we would be grateful if you would provide us with the location address or website name. Please contact us at **business@bpbonline.com** with a link to the material.

If you are interested in becoming an author

If there is a topic that you have expertise in, and you are interested in either writing or contributing to a book, please visit **www.bpbonline.com**. We have worked with thousands of developers and tech professionals, just like you, to help them share their insights with the global tech community. You can make a general application, apply for a specific hot topic that we are recruiting an author for, or submit your own idea.

Reviews

Please leave a review. Once you have read and used this book, why not leave a review on the site that you purchased it from? Potential readers can then see and use your unbiased opinion to make purchase decisions. We at BPB can understand what you think about our products, and our authors can see your feedback on their book. Thank you!

For more information about BPB, please visit **www.bpbonline.com**.

Join our book's Discord space

Join the book's Discord Workspace for Latest updates, Offers, Tech happenings around the world, New Release and Sessions with the Authors:

https://discord.bpbonline.com

Table of Contents

Chapter 1
Introducing PowerShell

Introduction

As is the tradition, we will take a few glimpses at the introduction to PowerShell, the history of PowerShell, look at what makes PowerShell so flexible and powerful, and introduce ourselves to the different landscapes of PowerShell integrations. This chapter addresses the following topics:

- A brief overview of what PowerShell is and why it is important.
- Explanation of the target audience for this chapter (for example, system administrators, developers, automation engineers, cloud administrators, database experts, and so on).
- Understand PowerShell integration choices.
- A brief history of the development and evolution of PowerShell.
- Overview of the key features in each version of PowerShell.
- Learn how to perform PowerShell installation and configuration.
- Explanation of how to use PowerShell ISE, Visual Studio Code, and Cloud Shell
- Learn why PowerShell is such a popular and widely used tool.

Figure 1.1 details the structure of the first chapter and provides a comprehensive introduction to PowerShell, covering the key concepts and features of the language, as well as theoretical details and examples for using it effectively:

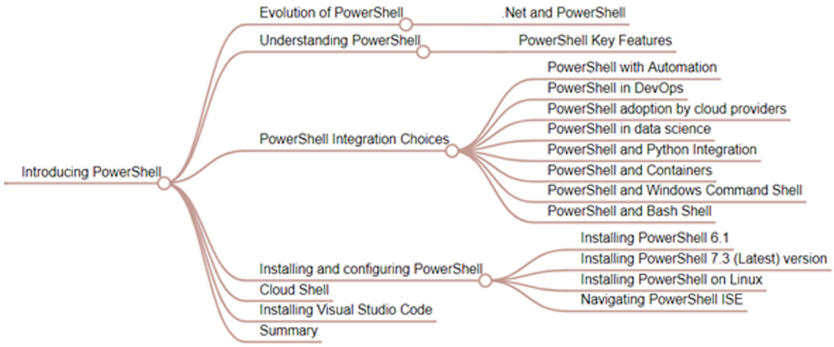

Figure 1.1: *Introducing PowerShell*

PowerShell has propelled in every way in the automation arena. Since the inception of PowerShell, it has become a de-facto tool for automation. It is a favorite solution of many Windows administrators—**capable of automating** almost any task in the Microsoft ecosystem.

If you are new, it is understandable that when we hear about new tools, we get overwhelmed with many fuzzy words. It is a collective effort to deliver the details seamlessly. In this book, you will explore an organized way to learn PowerShell.

Are you ready? So, why wait?

Let us kick off the PowerShell journey.

Evolution of PowerShell

PowerShell is a shell and scripting language developed by Microsoft. It has evolved over time and has become an essential tool for many IT professionals. Its robust and flexible feature set continues to be a leading tool for system administrators and IT professionals in the years to come.

Microsoft invented PowerShell, and its initial design and development was led by Jeffrey Snover, a Technical Fellow at Microsoft. Jeffrey Snover and his team started working on PowerShell in the early 2000s, with the first version of PowerShell (then known as "Monad") being released in 2006.

The primary goal of PowerShell was to provide a modern, object-oriented scripting and automation platform for Windows that could compete with the Unix shells and tools that were popular in the IT industry at the time. Microsoft recognized the need for a more robust and flexible tool for managing and automating Windows systems, and PowerShell was developed to meet this need.

PowerShell is an automation and configuration management framework developed by Microsoft. It was first introduced in 2006 as Windows PowerShell 1.0; it has evolved into a more versatile and powerful tool.

Figure 1.2 shows a brief overview of the evolution of PowerShell:

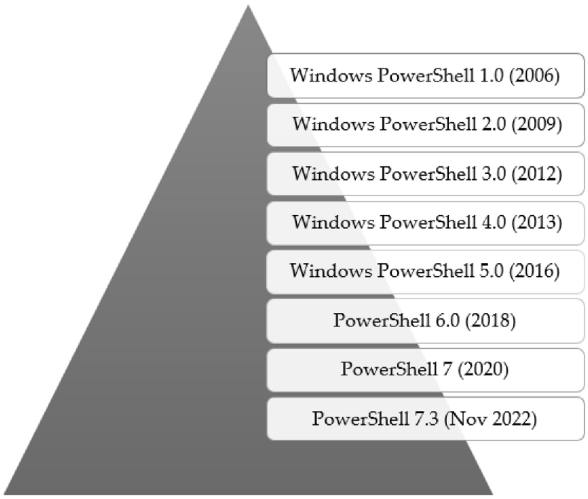

Figure 1.2: *Evolution of PowerShell*

- **Windows PowerShell 1.0 (2006):** This was the first release of PowerShell, and it was designed to be a command-line shell and scripting language for Windows. It had basic features like a cmdlet, pipeline, and remoting support.
- **Windows PowerShell 2.0 (2009):** This release included new features such as the introduction of background jobs, advanced functions, and transcriptions. It also improved remote capabilities, allowing for the remote management of multiple systems simultaneously.
- **Windows PowerShell 3.0 (2012):** This release included new features such as workflows, improved module management, and enhanced security features. It also improved performance and support for Windows Server 2012 and Windows 8.
- **Windows PowerShell 4.0 (2013):** This release included new features such as **Desired State Configuration** (**DSC**), which allowed administrators to define and maintain a desired state for their systems. It also had improved support for cloud services, such as Microsoft Azure.
- **Windows PowerShell 5.0 (2016):** This release included new features such as OneGet, which allowed for easy management of software packages and the ability to manage Linux systems from Windows. It also had improved support for .NET Core and Docker.
- **PowerShell 6.0 (2018):** It was released in 2018. This version was a significant update to PowerShell, focusing on cross-platform compatibility. It was released for Windows, macOS, and Linux, introducing support for running PowerShell on multiple platforms.

- **PowerShell 7 (2020):** This is the latest version of PowerShell, and it includes new features such as improved performance and support for .NET 5, improved error handling, and support for new platforms such as macOS and Linux.
- **PowerShell 7.3 (2022):** This was released in November 2022, the latest version of the shell and scripting language developed by Microsoft. It is a minor release that includes many bug fixes, performance improvements, and some new features and enhancements.

.NET and PowerShell

PowerShell is built on top of the .NET Framework or .NET Core, which provides the runtime environment and necessary libraries for PowerShell to execute its commands and scripts.

Different versions of PowerShell have additional requirements for the underlying .NET version. For example, PowerShell 1.0 and 2.0 require .NET Framework 2.0, PowerShell 5.1 requires .NET Framework 4.5.2, and PowerShell 7.0 requires .NET Core 3.1.

Suppose you try to run a version of PowerShell that requires a higher or lower version of .NET than what is currently installed on your system. In that case, you may encounter errors or unexpected behavior. Therefore, having the correct version of .NET installed is essential for PowerShell to function correctly.

One of the key reasons why PowerShell is more efficient than VBScript is its integration with the .NET Framework. PowerShell was designed from the ground up to be a modern scripting language that leverages the full power of the .NET Framework, including access to its vast library of pre-built code and runtime services.

Using .NET Classes in PowerShell Scripts provides the ability to reference .NET classes to perform advanced programming operations within scripts. For example, the class **`System.Math`** contains methods and properties for mathematical operations.

```
[System.Math]::PI - PI is a property and returns the value of Pi

[System.Math]::Sqrt(4) - sqrt is the method and returns the square root of 4

[System.Math]::Pow(3,2) - Pow is a method that returns the value of 3 to the power of 2.
```

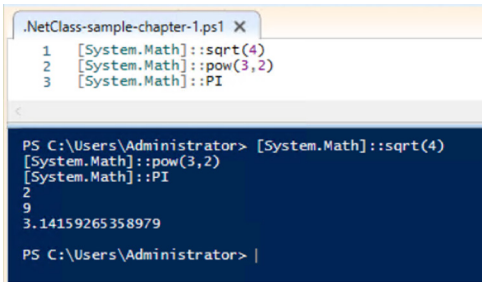

Figure 1.3: *.NET class sample*

PowerShell's verb-noun command format (cmdlet) was borrowed from the **Digital Command Language** (**DCL**) used on OpenVMS-based systems. The structure was chosen because it is more intuitive for IT administrators, who think of actions first, whereas programmers tend to think of objects and activities afterward. We will discuss more cmdlets in the upcoming chapter.

PowerShell key features

PowerShell is an automation and configuration management framework developed by Microsoft. It provides a powerful command-line interface for Windows, Linux, and macOS, allowing users to perform a wide range of tasks, such as managing files and directories, automating tasks, and managing and configuring systems.

Some key features of PowerShell include the following:

Cmdlets	PowerShell includes over 4,000 built-in cmdlets that allow you to perform various administrative tasks, such as managing files, registry entries, and process.
Piping	PowerShell supports piping, which allows you to pass the output of one cmdlet as input to another cmdlet. This enables you to build complex command chains to perform sophisticated administrative tasks.
Scripting	PowerShell provides a full-fledged scripting environment, with support for variables, loops, functions, and conditional statements. You can create reusable scripts to automate complex tasks and reduce manual effort.
Object-Oriented	PowerShell is object-oriented, which means that it works with objects, rather than text. This allows you to manipulate data more effectively and provides more insight into the data you are working with.
.NET Framework Integration	PowerShell is built on top of the .NET Framework, which provides a rich set of libraries for developers to use. This integration allows you to leverage the capabilities of the .NET Framework in your PowerShell scripts.
Remoting	PowerShell supports remoting, which allows you to run commands on remote computers. This makes it easy to manage multiple systems from a single location.
Modules and Provider Model	PowerShell provides a provider model that enables you to access data from a variety of sources, such as the file system, registry, and Active Directory. This provides a unified interface to access data, regardless of its location.
Integrated Help System	PowerShell includes an integrated help system that provides detailed information about cmdlets, scripts, and other elements of the shell. This makes it easier to learn and use PowerShell effectively.

Figure 1.4: *PowerShell key features*