Introduction

My introduction to PowerShell came from a webinar presented by Pragmatic Works' consultant and Microsoft MVP, Robert Cain. I was amazed by PowerShell but did not have an immediate application for it. Then, I started on a contract to migrate a client from a cloud architecture to a hybrid of cloud and on-premises SQL Server. The ETL tool was not SQL Server Integration Services, and it had a number of limitations. We needed to load external files, which required a good deal of pre-processing. For example, we needed to decrypt the files and unzip them. In some cases, we needed to add a header row with column names. Multiple files had to be consolidated into one file before loading. After processing, the files had to be archived. The client had already started coding these tasks as legacy-style batch files. The deadlines were tight, and the client did not want any delays. As I reviewed the batch files, with their cryptic coding, I knew investing in this outdated technology was the wrong way to go. I convinced them to let me rewrite the scripts in PowerShell.

It was a challenge to learn a new language from scratch while developing a solution with frequent deliverables, but it turned out to be the right decision. The final solution was scalable, extensible, and reusable. Most important, my passion for PowerShell started.

Before long, I started doing PowerShell presentations at PASS chapter meetings, Code Camps, and SQL Saturdays. The more I learned about PowerShell, the more excited I became. However, I found it difficult to find books or blogs that did anything but Windows Administration tasks. I used bits and pieces of these to cobble together what I needed. I was surprised that no one had written a book that covered application or database development. Professional colleagues agreed this was lacking, which kept many from adopting PowerShell. Maybe I could step up and fill in the gap?

A colleague from my FM Global days, Grant Fritchey, had written several successful books on SQL Server, so I asked him if he would refer me to a publisher. Grant recommended me to Jonathan Gennick at Apress, which eventually led to this book. This is a labor of love. I worked hard to make the examples complex and real world enough to be useful. This poses a challenge when making the scripts work in any environment without change, but I did my best. PowerShell was originally intended to be a Windows Administration tool, and many will adamantly insist that is all it should be used for. However, I have found PowerShell to be as capable as any language, but richer in features and far less verbose. I don't recommend abandoning C# or VB.NET, but I do think there are applications where PowerShell is a better choice. In the end, PowerShell is another tool in your belt.

Why This Book?

This is the book I wish 0'dI had when I started learning PowerShell. Once I got beyond the preliminaries and the "Hello World" scripts, I found a dearth of material to go further. I searched the Internet for examples of advanced scripting, but rarely found more than a few lines, and even that was focused on Windows Administration. There are many PowerShell books available, but I found these to also be focused on administration tasks. A few books covered using PowerShell to create utilities to aid in development, but I could not find any that presented PowerShell as a development language or discussed advanced programming. For example, there are many blogs on how to retrieve data from SQL Server, but what about manipulating the data and writing it back? Professional applications need to isolate items that can change, such as server names, folder names, and credentials, so they can be easily reconfigured from one

environment to the next. But I could not find any books or blogs on this subject. I found short explanations on how to create function libraries called modules, but none that explained how to include logic that controlled how the module loaded. While I found some simple examples of creating custom PowerShell objects, I could not find any that included methods that could accept parameters. This book covers all these things and more!

A challenge to learning PowerShell is that there is so much hype that no one wants to point out its bugs and limitations. Even the online Microsoft documentation sometimes omits what a command or parameter cannot do. An example this is the documentation on PowerShell transactions. The documentation leads you to think this feature is far more powerful than it is. Another potential challenge, especially to database professionals, is what PowerShell calls `nulls`. I found this to be particularly confusing and included a detailed discussion about it in the book. By reading this book, you can save many hours searching the Internet to resolve issues and questions.

Whether or not you agree that PowerShell should be used for application development, I think the language must be treated as such in order to truly see its capabilities and limitations. To summarize, the purpose of this book is to present PowerShell as a development platform and demonstrate many techniques toward this end.

Who Is This Book For?

This book is for the professional who wants to go beyond the basics and master PowerShell development. It is not intended as a reference or as a beginner guide, although it does include some background and an introduction to the basics. The ideal reader has some experience in a programming language and its related constructs, such as variables, loops, conditional expressions, etc. It helps to have a degree of comfort reading code. While most of the examples focus on data-orientated tasks, the techniques are universally applicable to professional development. Application developers and database developers will probably get the most out of this book, but anyone willing to learn and develop their PowerShell programming skills will benefit.

How to Read This Book

This book starts with the basics and develops more-advanced concepts with each chapter. It is recommended that the reader start at the beginning and read the chapters sequentially. The scripts tend to build on each other as the book progresses and often reference prior chapters. This is necessary to lay the groundwork for the truly advanced topics such as using PowerShell for ETL and developing workflows. I find the best way to learn a language is with examples. With that in mind, the approach used here is to provide a complete listing first and then walk through it a few lines at a time and explain exactly how it works.

System Requirements

The code in this book was developed and tested using:

- PowerShell 4.0
- Windows 7 SP1.
- SQL Server 2012
- Adventure Works 2008 (or later should work with those examples)
- Office 2013
- Internet Explorer 11
- PostgreSQL 9.3

About the Files…

You can download example files for this book from the Apress.com catalog page at:

http://www.apress.com/9781484205426

Look for the tab or link labeled Source Code/Downloads. The examples will be in a .zip archive. Download that archive. Unzip it. Begin at the readme file.

Note: Formatting of the code listings in the book was a challenge, because PowerShell does not allow a statement to continue onto another line without a special continuation character. As much as possible, these line continuation characters were added so the book matches the script files, but in some cases the line in the book may wrap. If you see a difference, the script file is the correct one.