Windows PowerShell ステップアップ講座

by むたぐち（牟田口大介）
Microsoft MVP
for Data Center Management -
Admin Frameworks

PowerShellとは

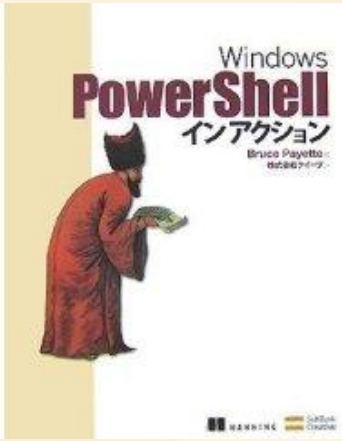
- Windows PowerShellとは
- .NET Frameworkをベースに動作する、Windowsの新しいシステム管理用シェル＆スクリプト実行環境
- Windows Server2003/XP/Vista用がダウンロード可能。Server 2008には標準搭載
- ver 2.0 CTPも出ました

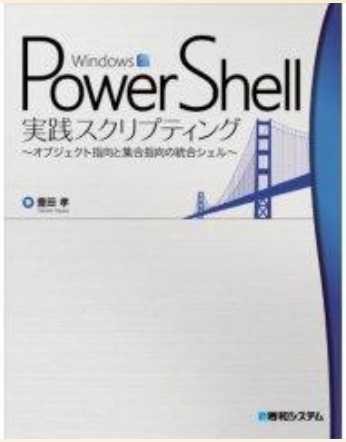
PowerShellの情報源（Web）

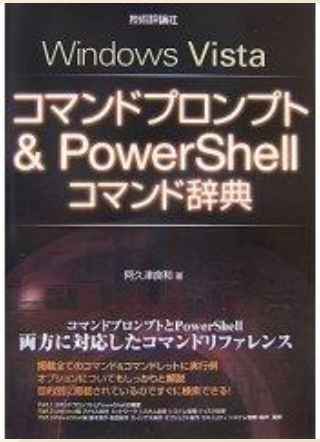
- Windows PowerShell でのスクリプティング
  http://www.microsoft.com/japan/technet/scriptcenter/hubs/msh.mspx
- PowerShell Memo
  http://d.hatena.ne.jp/newpops/
- Shigeya Tanabe's blog
  http://blogs.technet.com/stanabe/default.aspx
- その他紹介記事、初心者向け記事数点（@IT、ITPro、codezineなど）
- PowerShell Scripting（実質リンク集）
  http://www.roy.hi-ho.ne.jp/mutaguchi/powershell/
- Scripting Weblog
  http://blogs.wankuma.com/mutaguchi/

PowerShellの情報源（書籍）

- 雑誌
  - Windows Server World
    http://www.windows-world.jp/
- 書籍

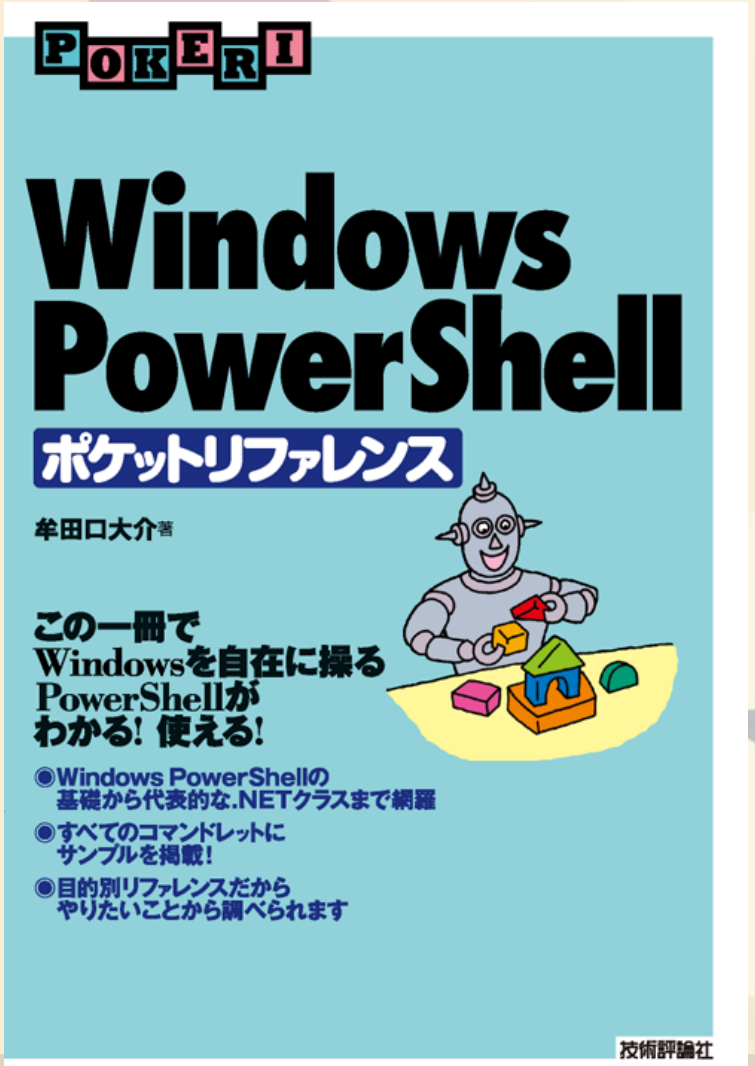






- そして2008年4月…

Windows PowerShell ポケットリファレンス4/11発売予定



- 発売日：2008/4/11
- 価格：￥ 2,079（税込）
- 出版社：技術評論社
- ページ数：392P
- ISBN：978-4774134109
- 著者：牟田口大介
- http://winscript.jp/

PowerShellポケットリファレンスの特徴

- PowerShellの使い方、文法解説（第一部100P）、コマンドレット（第二部200P）、.NETクラス・構造体（第三部100P）の三部構成。
- プログラミング初心者からUNIXシェルを使いこなしている方まで対応。
- コマンドレット章には実行サンプルと、多くにスクリプトサンプルを掲載。→今回はそこからピックアップしてデモをします。

PowerShellの基本・コマンドレット

- コマンドプロンプトの内部コマンド(cdとかdirとか)に相当する129種のコマンドレット（cmdlet）を組み合わせて使うのが基本です。
- 大きく分けてPSドライブ操作、PSユーティリティ、システム管理機能呼び出し、オブジェクトの操作の4種類（むたぐち分類法）。
- コマンドレットの引数（パラメータ）も戻り値もみな.NETのオブジェクトである。だからメソッドを呼び出したりもできる。

コマンドレットの分類表

・PSドライブ操作

項目の内容を操作するには
Add-Content
Clear-Content
Get-Content
Set-Content

項目を操作するには
Get-ChildItem
Clear-Item
Copy-Item
Get-Item
Move-Item
New-Item
Remove-Item
Rename-Item
Set-Item

プロパティ操作を行うには
Clear-ItemProperty
Copy-ItemProperty
Get-ItemProperty
Move-ItemProperty
New-ItemProperty
Remove-ItemProperty
Rename-ItemProperty
Set-ItemProperty

項目を実行するには
Invoke-Item

PSドライブを扱うには
Get-PSDrive
Get-PSProvider
New-PSDrive
Remove-PSDrive

ロケーションを取得・設定するには
Get-Location
Set-Location
Push-Location
Pop-Location

パスの操作を行うには
Convert-Path
Resolve-Path
Join-Path
Split-Path
Test-Path

ACLの取得・設定を行うには
Get-Acl
Set-Acl

・PSユーティリティ

ヘルプを取得するには
Get-Help

コマンドレットの一覧を取得するには
Get-Command

ファイルまたは文字列から指定の文字列を抜き出すには
Select-String

変数の値を操作するには
Get-Variable
Set-Variable
New-Variable
Remove-Variable
Clear-Variable

履歴を参照・追加・実行するには
Add-History
Get-History
Invoke-History

エイリアスを操作するには
Get-Alias
Set-Alias
New-Alias
Export-Alias
Import-Alias

メッセージを書き込むには
Write-Host
Write-Output
Write-Verbose
Write-Warning
Write-Debug
Write-Error

文字列の入力を読み取るには
Read-Host

暗号化文字列・セキュア化された文字列を扱うには
ConvertFrom-SecureString
ConvertTo-SecureString

プログレスバーを表示するには
Write-Progress

シェル・スクリプトを中断するには
Start-Sleep

デバッグを行うには
Set-PSDebug
Start-Transcript
Stop-Transcript
Get-TraceSource
Set-TraceSource
Trace-Command

コマンドの実行時間を計測するには
Measure-Command

文字列をコマンドとして実行するには
Invoke-Expression

ホストオブジェクトを参照するには
Get-Host

設定ファイルを読み込むには
Update-FormatData
Update-TypeData

実行ポリシー取得・設定するには
Get-ExecutionPolicy
Set-ExecutionPolicy

スクリプトファイルの署名を取得・設定するには
Get-AuthenticodeSignature
Set-AuthenticodeSignature

Get-PfxCertificate

PSスナップインを操作するには
Get-PSSnapin
Add-PSSnapin
Remove-PSSnapin
Export-Console

・管理機能呼び出し

日付時刻を取得・設定するには
Get-Date
Set-Date

TimeSpanオブジェクトを作成するには
New-TimeSpan

サービスを操作するには
Get-Service
Start-Service
Stop-Service
Restart-Service
Suspend-Service
Resume-Service
Set-Service
New-Service

WMIのオブジェクトを扱うには
Get-WmiObject

資格情報を取得するには
Get-Credential

プロセスを取得・停止するには
Get-Process
Stop-Process

イベントログの情報を取得するには
Get-EventLog

カルチャ、UIカルチャに関する情報を取得するには
Get-Culture
Get-UICulture

・オブジェクト操作

オブジェクトをフィルタ・加工するには
Where-Object
Tee-Object
Sort-Object
Select-Object
Get-Unique
Group-Object

.NETかCOMのオブジェクトを作成するには
New-Object

オブジェクトを計測するには
Measure-Object

オブジェクトのメンバを列挙するには
Get-Member

オブジェクトにメンバを追加するには
Add-Member

オブジェクトの比較を行うには
Compare-Object

オブジェクトを列挙するには
ForEach-Object

オブジェクトの出力の形式を変更するには
Format-List
Format-Table
Format-Wide
Format-Custom

オブジェクトを様々なデバイス、ファイル、文字列に出力するには
Out-Default
Out-Host
Out-File
Out-Null
Out-Printer
Out-String

オブジェクトをファイルに出力・ファイルから入力するには
Export-Clixml
Import-Clixml
Export-Csv
Import-Csv
ConvertTo-Html

コマンドレットの使い方

- もっとも単純な例
  - PS >【コマンドレット名】 ex) Get-Date
- パラメータを取る場合
  - PS >【コマンドレット名】【-パラメータ名】 ex) Get-ChildItem -force
- パラメータと値を取る場合
  - PS >【コマンドレット名】【-パラメータ名】【パラメータ】
  - ex) Get-Command -type Cmdlet
- 複数パラメータを取る場合
  - PS >【コマンドレット名】【-パラメータ名1】【パラメータ1 】【-パラメータ名2】【パラメータ2】 ex) Get-Eventlog -LogName system -Newest 5
- パラメータ名を省略した場合
  - PS >【コマンドレット名】【パラメータ1 】【パラメータ2】
  - ex) Rename-Item a.txt b.txt

コマンドレットの組み合わせ　基本編

- コマンドレットは単体で使ってもいいが、組み合わせて使うとよりおいしい。
- 組み合わせ方法
  1. オブジェクトが渡るパイプを使いコマンドレットを連結
     - Get-Process | Sort-Object handles | Format-List | Out-Host -paging
  2. 変数を使いコマンドレットの戻り値をパラメータに
     - $name = Read-Host "Input Your Name"
       Write-Host -object $name
  3. コマンドレットを別の物のパラメータに直接与える
     - Split-Path -path (Get-Location) -leaf

コマンドレットの組み合わせ　応用編

- コマンドレットの戻り値が.NETオブジェクトということを意識し、メソッドやプロパティを呼ぶ
  4. コマンドレットの戻り値のメソッド・プロパティを用いる

```
(Get-Date).AddDays(30)
```

or

```
$now=Get-Date
$now.AddDays(30)
```

  5. フィルタスクリプトを用いる

```
Get-ChildItem *.ps1 | Where-Object {$_.Length -gt 1kb}
```

whereあるいは?というエイリアスが使用可

  6. 列挙スクリプトを用いる

```
Get-ChildItem -recurse | ForEach-Object{$_.FullName}
```

foreachあるいは%というエイリアスが使用可

関数・スクリプト化

- 何回も同じ処理をする場合はプロファイルに関数として保存するか、拡張子ps1ファイルにして保存することで、何回でも呼び出し可能。
- 今回は諸般の都合でスクリプト化しています。
- 関数もスクリプトもfunction func{}あるいは script.ps1中で

  param($param1)
- とすることで

  func -param1 valや.¥script.ps1 -param1 val

  のように呼び出せます。($param1に"val"が格納)

デモ

以上を踏まえてあとは具体的なスクリプトサンプルを
ステップを踏みながら見て行きましょう。

DEMO

まとめ

- コマンドレットは単体で使っても組み合わせて使ってもおいしい。
- .NET FrameworkのオブジェクトをNew-Objectコマンドレットで呼び出して使わなくてもかなりのことが他のコマンドレットの組み合わせでできる。
- PowerShellはシェルであると同時に強力なスクリプト言語でもある。