



# Windows PowerShell 逆引きハンドブック

蒲生 瞳男◆著

目的から **PowerShell** の  
コマンドレットが引ける  
逆引きリファレンスの決定版!

Windowsシステム/サーバー管理者

# 必携書!

バージョン  
5.0/4.0/  
3.0/2.0  
対応!



24時間無料でサンプルデータをダウンロードできます。

 C&R研究所

## C&R研究所について

C&R研究所は新潟市にある出版社です。ユニークな社風や教育方針は新聞やテレビなどで紹介されたりします。詳細については、次のWebサイトでご覧いただくことができます。

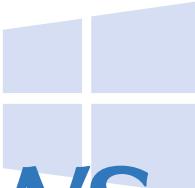
**[www.c-r.com](http://www.c-r.com)**

また、新潟本社には2代目会社犬「ラッキー」がいます。名刺を持つ正式な社員として広報部に勤務しつつ、セラピードッグとして社内のメンタルヘルスにも貢献しています。



●2代目会社犬「ラッキー」

---



# Windows PowerShell

## 逆引きハンドブック

---

蒲生 瞳男◆著

## ■権利について

- 本書に記述されている社名・製品名などは、一般に各社の商標または登録商標です。
- 本書では™、©、®は割愛しています。

## ■本書の内容について

- 本書は著者・編集者が実際に操作した結果を慎重に検討し、著述・編集しています。ただし、本書の記述内容に関わる運用結果にまつわるあらゆる損害・障害につきましては、責任を負いませんのであらかじめご了承ください。
- 本書で紹介している操作の画面は、Windows 10を基本にしています。他の環境では、画面のデザインや操作が異なる場合がございますので、あらかじめご了承ください。
- 本書は2016年2月現在の情報で記述しています。

## ■サンプルについて

- 本書で紹介しているサンプルは、C&R研究所のホームページ (<http://www.c-r.com>) からダウンロードすることができます。ダウンロード方法については、5ページを参照してください。
- サンプルデータの動作などについては、著者・編集者が慎重に確認しております。ただし、サンプルデータの運用結果にまつわるあらゆる損害・障害につきましては、責任を負いませんのであらかじめご了承ください。
- サンプルデータの著作権は、著者及びC&R研究所が所有します。許可なく配布・販売することは堅く禁止します。

### ●本書の内容についてのお問い合わせについて

この度はC&R研究所の書籍をお買いあげいただきましてありがとうございます。本書の内容に関するお問い合わせは、「書名」「該当するページ番号」「返信先」を必ず明記の上、C&R研究所のホームページ(<http://www.c-r.com/>)の右上の「お問い合わせ」をクリックし、専用フォームからお送りいただくか、FAXまたは郵送で次の宛先までお送りください。お電話でのお問い合わせや本書の内容とは直接的に関係のない事柄に関するご質問にはお答えできませんので、あらかじめご了承ください。

〒950-3122 新潟県新潟市北区西名目所4083-6 株式会社 C&R研究所 編集部  
FAX 025-258-2801  
「Windows PowerShell逆引きハンドブック」サポート係

PowerShell(Windows PowerShell)は、Windows Server 2008 R2、Windows 7から標準で搭載されているシステム管理用のコマンドシェルです。シェルとしてだけでなく、スクリプト言語としても利用できる、とてもパワフルで便利なツールです。

本書は、PowerShellを使ってシステム管理を行う人のためのリファレンスです。PowerShellのコマンドレットを目的別に分類し、やりたいことから探していただけるように、逆引き形式を採用しています。手元に置いていつでもお使いいただけるようにA5サイズを採用していますので、何度も繰り返しページをめくってみてください。

本書は、PowerShellのバージョン2.0からバージョン5.0の基本的なコマンドレットの実行例を、初心者にもわかりやすいシンプルな事例で紹介しています。さらに、ONEPOINTではポイントとなるパラメータ(引数)についてわかりやすい解説とともに、COLUMNには、読者のみなさんの理解を深める手助けとなるように、補足情報を豊富に盛り込んでいます。また、コマンドレットの構文も掲載しています。

PowerShellは、日本語のドキュメントが少なく、バージョン3.0以降ではヘルプも日本語化されていません。このため、英語が苦手な人にとっては、習得が困難です。また、公開されているドキュメントの記述も、間違っていたり、情報が不足していたりと、十分ではありません。

本書は、各コマンドレットを実際に実行した結果と、ドキュメントやヘルプの記述を比較し、相違点や矛盾点を精査して記述していますので、これからPowerShellを学ぶ初心者はもちろん、PowerShellのパワーユーザーの参考書としても、最適な内容となっています。

最後に、本書を執筆にするにあたってお世話になったスタッフの皆さんに心から感謝を申し上げます。そして、本書がPowerShellでシステム管理を行う読者の皆様に少しでもお役に立てれば、これ以上の幸せはありません。

2016年2月

蒲生 隆男

# 本書について

## III 本書の表記方法

本書の表記についての注意点は、次のようにになります。

### ▶ コマンドの実行例について

本書のコマンドの実行例については、Windows 10のバージョン5.0を基本にしています。カレントディレクトは「C:¥work」を基本にしています。他のOSやバージョンによって実行結果などが異なる場合があります。また、誌面の都合上、折り返し位置などが実際の画面と異なる場合があります。あらかじめ、ご了承ください。

なお、白地に色付きの文字については、コマンドや実行結果についての解説になります。実際の実行例では表示されません。

### ▶ バージョン5.0の64bit版について

執筆時点では、バージョン5.0の64bit版では、パスに日本語が含まれている、または日本語を入力する場合、フォントをMSゴシックに変更しないと、カーソルの位置などがずれるという問題があります。

この問題は、フォントをMSゴシックに設定(タイトルバーのアイコンをクリックしてメニューから[プロパティ(P)]を選択し、「フォント」タブの[フォント(F)]で「MS ゴシック」を選択して[OK]ボタンをクリック)すると解決します。

### ▶ バージョンの表記について

本文中では、バージョン2.0/3.0/4.0/5.0を、それぞれV2/V3/V4/V5と表記しています。

### ▶ コマンドの中の団について

本書に記載したコマンドの実行例の中の団は、コマンドの入力を確定するためなどの[Enter]キーを押すことを表しています。

### ▶ 行継続のプロンプトについて

本書に記載したコマンドの実行例の中の行頭の「>>」は、行継続のプロンプトを表しています。なお、バージョン2.0～4.0や5.0の32bit版は、最後の行を入力した後に、さらに「>>」が表示されるので[Enter]キーを押してください。

### ▶ コマンドやコードの中の■について

本書に記載したコマンドの実行結果やコードは、誌面の都合上、ページをまたがって記載されていることがあります。その場合は■の記号で、1つの実行結果・コードであることを表しています。

### ▶ 各項目の対応バージョンについて

各項目の先頭にはその項目に対応しているバージョンを記載しています。薄く表示されているマークは非対応を表しています。

### III サンプルファイルのダウンロードについて

本書のサンプルデータは、C&R研究所のホームページからダウンロードすることができます。  
本書のサンプルを入手するには、次のように操作します。

- ① 「<http://www.c-r.com/>」にアクセスします。
- ② トップページ左上の「商品検索」欄に「174-0」と入力し、[検索]ボタンをクリックします。
- ③ 検索結果が表示されるので、本書の書名のリンクをクリックします。
- ④ 書籍詳細ページが表示されるので、[サンプルデータダウンロード]ボタンをクリックします。
- ⑤ 下記の「ユーザー名」と「パスワード」を入力し、ダウンロードページにアクセスします。
- ⑥ 「サンプルデータ」のリンク先のファイルをダウンロードし、保存します。

サンプルのダウンロードに必要な  
ユーザー名とパスワード

ユーザー名 **psgr**  
パスワード **5xf7w**

※ユーザー名・パスワードは、半角英数字で入力してください。また、「J」と「j」や「K」と「k」などの大文字と小文字の違いもありますので、よく確認して入力してください。

サンプルはZIP形式で圧縮してありますので、解凍してお使いください。なお、サンプルは、コマンドを実行する際にデータなどが必要となる場合のファイルになります。使い方などについては、サンプルが必要な項目のHINTをご確認ください。

# CONTENTS

## CHAPTER 01 PowerShellの基礎知識

□□ 1	PowerShellとは	30
□□ 2	PowerShellのインストール	32
□□ 3	PowerShellの起動	34
□□ 4	コマンドレットについて	38
□□ 5	コマンドレットのパラメータについて	39
□□ 6	共通パラメータ/リスク軽減パラメータについて	42
□□ 7	エイリアスについて	46
□□ 8	リダイレクトについて	47
□□ 9	パイプラインについて	49
□ 1 □	ワイルドカードについて	51
□ 1 1	スクリプトについて	52
□ 1 2	プロバイダーについて	53
□ 1 3	リモート機能について	69
□ 1 4	自動変数とユーザー設定変数について	72

## CHAPTER 02 PowerShellの基本文法

□ 1 5	変数について	76
□ 1 6	型について	78
□ 1 7	文字列について	80
□ 1 8	配列について	83
□ 1 9	ハッシュテーブルについて	85
□ 2 □	演算子について	87
COLUMN ■特殊演算子について		
□ 2 1	条件分岐について	94
□ 2 2	ループ(繰り返し処理)について	95
□ 2 3	関数について	98
□ 2 4	関数のパラメータについて	102
□ 2 5	高度な関数について	104
□ 2 6	高度な関数のパラメータについて	106
□ 2 7	例外処理について	112
□ 2 8	その他の知っておきたい基本文法	113

## CHAPTER 03 ヘルプ・コマンド・エイリアス

□ 029 ヘルプを更新する .....	116
ONEPOINT■ヘルプを更新するには「Update-Help」を使用する	
COLUMN■ダウンロード制限を解除するには	
COLUMN■UIカルチャを指定して更新するには	
□ 030 ヘルプを保存する .....	118
ONEPOINT■ヘルプを保存するには「Save-Help」を使用する	
COLUMN■ダウンロードしたヘルプをインストールするには	
□ 031 ヘルプを表示する .....	120
ONEPOINT■ヘルプを表示するには「Get-Help」を使用する	
COLUMN■概念説明のヘルプトピックを表示するには	
COLUMN■ヘルプを表示するその他のコマンド	
□ 032 入力したコマンドの履歴を表示する .....	122
ONEPOINT■入力したコマンドの履歴を表示するには「Get-History」を使用する	
COLUMN■取得開始エントリや取得エントリ数を指定するには	
COLUMN■記録するコマンド数を変更するには	
□ 033 以前に入力したコマンドを再実行する .....	124
ONEPOINT■以前に入力したコマンドを再実行するには「Invoke-History」を使用する	
COLUMN■連続して以前のコマンドを複数実行するには	
□ 034 コマンドの履歴を追加する .....	126
ONEPOINT■コマンドの履歴を追加するには「Add-History」を使用する	
□ 035 コマンドの履歴を削除する .....	128
ONEPOINT■コマンドの履歴を削除するには「Clear-History」を使用する	
□ 036 コマンドを取得する .....	129
ONEPOINT■コマンドを取得するには「Get-Command」を使用する	
COLUMN■指定した名前のコマンドのみを取得するには	
COLUMN■指定された動詞/名詞を名前の一部とするコマンドレットを取得するには	
COLUMN■特定のコマンドの構文を表示するには	
□ 037 コマンドウィンドウを表示する .....	131
ONEPOINT■コマンドウィンドウを表示するには「Show-Command」を使用する	
COLUMN■コマンドウィンドウの高さと幅を指定するには	
COLUMN■エラーウィンドウにエラーを表示するには	
□ 038 エイリアスを取得する .....	134
ONEPOINT■エイリアスを取得するには「Get-Alias」を使用する	
COLUMN■指定したコマンドのエイリアスを取得するには	
□ 039 エイリアスを作成する .....	136
ONEPOINT■エイリアスを作成するには「New-Alias」を使用する	
□ 040 エイリアスを再設定する .....	138
ONEPOINT■エイリアスを再設定するには「Set-Alias」を使用する	
□ 041 エイリアスを保存する .....	139
ONEPOINT■エイリアスを保存するには「Export-Alias」を使用する	

□ 4.2 エイリアスをインポートする	141
ONEPOINT ■ エイリアスをインポートするには「Import-Alias」を使用する	
□ 4.3 文字列をコマンドとして実行する	142
ONEPOINT ■ 文字列をコマンドとして実行するには「Invoke-Expression」を使用する	
□ 4.4 コマンドの実行時間を計測する	143
ONEPOINT ■ コマンドの実行時間を計測するには「Measure-Command」を使用する	

## CHAPTER 04 日付/時刻・変数・文字列

□ 4.5 日付/時刻を取得する	146
ONEPOINT ■ 日付/時刻を取得するには「Get-Date」を使用する	
COLUMN ■ 日時の書式を設定するには	
□ 4.6 システム日時を設定する	148
ONEPOINT ■ システム日時を設定するには「Set-Date」を使用する	
COLUMN ■ 時間を調整するには	
□ 4.7 日付計算を行う	150
ONEPOINT ■ 日付計算を行うには「New-TimeSpan」を使用する	
COLUMN ■ 日時の加算/減算を行うには	
□ 4.8 変数を取得する	152
ONEPOINT ■ 変数を取得するには「Get-Variable」を使用する	
□ 4.9 新しい変数を作成する	154
ONEPOINT ■ 新しい変数を作成するには「New-Variable」を使用する	
COLUMN ■ 変数のスコープやオプションを設定するには	
□ 5.0 変数の値を設定する	156
ONEPOINT ■ 変数の値を設定するには「Set-Variable」を使用する	
□ 5.1 変数の値を削除する	157
ONEPOINT ■ 変数の値を削除するには「Clear-Variable」を使用する	
□ 5.2 変数を削除する	158
ONEPOINT ■ 変数を削除するには「Remove-Variable」を使用する	
□ 5.3 メッセージを表示する	160
ONEPOINT ■ メッセージを表示するには「Write-Host」を使用する	
COLUMN ■ セパレータを指定するには	
□ 5.4 詳細メッセージを表示する	162
ONEPOINT ■ 詳細メッセージを表示するには「Write-Verbose」を使用する	
□ 5.5 エラーメッセージを表示する	163
ONEPOINT ■ エラーメッセージを表示するには「Write-Error」を使用する	
□ 5.6 進行状況を表すバーを表示する	165
ONEPOINT ■ 進行状況を表すバーを表示するには「Write-Progress」を使用する	
□ 5.7 デバッグメッセージを表示する	166
ONEPOINT ■ デバッグメッセージを表示するには「Write-Debug」を使用する	

□ 5.8 オブジェクトを表示する	167
<b>ONEPOINT</b> ■オブジェクトを表示するには「Write-Output」を使用する	
□ 5.9 警告メッセージを表示する	168
<b>ONEPOINT</b> ■警告メッセージを表示するには「Write-Warning」を使用する	
□ 6.0 情報メッセージを表示する	169
<b>ONEPOINT</b> ■情報メッセージを表示するには「Write-Information」を使用する	
□ 6.1 コマンドプロンプトからの文字入力を読み取る	170
<b>ONEPOINT</b> ■コマンドプロンプトからの文字入力を読み取るには 「Read-Host」を使用する	
<b>COLUMN</b> ■セキュリティで保護された文字列を作成するには	
□ 6.2 セキュア文字列を暗号化文字列に変換する	172
<b>ONEPOINT</b> ■セキュア文字列を暗号化文字列に変換するには 「ConvertFrom-SecureString」を使用する	
<b>COLUMN</b> ■Rijndael暗号化アルゴリズムを使用して暗号化文字列に変換するには	
□ 6.3 暗号化文字列をセキュア文字列に変換する	174
<b>ONEPOINT</b> ■暗号化文字列をセキュア文字列に変換するには 「ConvertTo-SecureString」を使用する	
<b>COLUMN</b> ■プレーンテキストをセキュア文字列に変換するには	
□ 6.4 テキストを検索する	176
<b>ONEPOINT</b> ■テキストを検索する「Select-String」を使用する	
<b>COLUMN</b> ■ファイルのテキストを検索するには	
□ 6.5 XMLを検索する	178
<b>ONEPOINT</b> ■XMLを検索するには「Select-XML」を使用する	
<b>COLUMN</b> ■XML文字列を検索するには	
□ 6.6 文字列データをハッシュテーブルに変換する	180
<b>ONEPOINT</b> ■文字列データをハッシュテーブルに変換するには 「ConvertFrom-StringData」を使用する	
□ 6.7 スクリプトを多言語対応にする	181
<b>ONEPOINT</b> ■スクリプトを多言語対応にするには「Import-LocalizedData」を使用する	
<b>COLUMN</b> ■psd1ファイルについて	
□ 6.8 文字列を分割する	184
<b>ONEPOINT</b> ■文字列を分割するには「ConvertFrom-String」を使用する	
<b>COLUMN</b> ■プロパティに任意の名前を付けるには	
<b>COLUMN</b> ■意図するデータだけを抽出するには	

## CHAPTER □5 アイテム・ファイル

□ 6.9 アイテムを取得する	188
<b>ONEPOINT</b> ■アイテムを取得するには「Get-Item」を使用する	
□ 7.0 アイテムの値を設定する	190
<b>ONEPOINT</b> ■アイテムの値を設定するには「Set-Item」を使用する	

□ 7.1	新しいアイテムを作成する	192
ONEPOINT	■新しいアイテムを作成するには「New-Item」を使用する	
COLUMN	■アイテム名の指定を省略するには	
□ 7.2	アイテムをコピーする	194
ONEPOINT	■アイテムをコピーするには「Copy-Item」を使用する	
COLUMN	■ディレクトリの内容全体をコピーするには	
□ 7.3	アイテムを移動する	196
ONEPOINT	■アイテムを移動するには「Move-Item」を使用する	
COLUMN	■ディレクトリの内容を移動するには	
□ 7.4	アイテムの名前を変更する	198
ONEPOINT	■アイテムの名前を変更するには「Rename-Item」を使用する	
□ 7.5	アイテムの値を削除する	199
ONEPOINT	■アイテムの値を削除するには「Clear-Item」を使用する	
□ 7.6	アイテムを削除する	201
ONEPOINT	■アイテムを削除するには「Remove-Item」を使用する	
COLUMN	■ディレクトリの全体を削除するには	
□ 7.7	アイテムの既定のアクションを実行する	203
ONEPOINT	■アイテムの既定のアクションを実行するには「Invoke-Item」を使用する	
□ 7.8	アイテムの内容を取得する	205
ONEPOINT	■アイテムの内容を取得するには「Get-Content」を使用する	
COLUMN	■取得行数を指定するには	
COLUMN	■配列表記で内容を参照するには	
□ 7.9	アイテムに内容を追加する	208
ONEPOINT	■アイテムに内容を追加するには「Add-Content」を使用する	
□ 8.0	アイテムの内容を変更する	210
ONEPOINT	■アイテムの内容を変更するには「Set-Content」を使用する	
□ 8.1	アイテムの内容を削除する	212
ONEPOINT	■アイテムの内容を削除するには「Clear-Content」を使用する	
□ 8.2	アイテムのプロパティを取得する	214
ONEPOINT	■アイテムのプロパティを取得するには「Get-ItemProperty」を使用する	
COLUMN	■指定したプロパティを取得するには	
COLUMN	■レジストリの値とデータを取得するには	
□ 8.3	アイテムのプロパティを設定する	217
ONEPOINT	■アイテムのプロパティを設定するには「Set-ItemProperty」を使用する	
COLUMN	■レジストリの値とデータを設定するには	
□ 8.4	アイテムのプロパティを作成する	220
ONEPOINT	■アイテムのプロパティを作成するには「New-ItemProperty」を使用する	
□ 8.5	アイテムのプロパティをコピーする	222
ONEPOINT	■アイテムのプロパティをコピーするには 「Copy-ItemProperty」を使用する	

□ 8.6 アイテムのプロパティを移動する	224
ONEPOINT■	アイテムのプロパティを移動するには「Move-ItemProperty」を使用する
□ 8.7 アイテムのプロパティ名を変更する	226
ONEPOINT■	アイテムのプロパティ名を変更するには「Rename-ItemProperty」を使用する
□ 8.8 アイテムのプロパティ値をクリアする	228
ONEPOINT■	アイテムのプロパティ値をクリアするには「Clear-ItemProperty」を使用する
□ 8.9 アイテムのプロパティを削除する	230
ONEPOINT■	アイテムのプロパティを削除するには「Remove-ItemProperty」を使用する
□ 9.0 アイテムのプロパティ値を取得する	232
ONEPOINT■	アイテムのプロパティ値を取得するには「Get-ItemPropertyValue」を使用する
□ 9.1 アイテムの子アイテムを取得する	234
ONEPOINT■	アイテムの子アイテムを取得するには「Get-ChildItem」を使用する
COLUMN■	すべての子コンテナー内のアイテムを取得するには
□ 9.2 ファイルを圧縮する	236
ONEPOINT■	ファイルを圧縮するには「Compress-Archive」を使用する
COLUMN■	特定の種類のファイルをまとめて圧縮するには
COLUMN■	既存の圧縮ファイルを更新するには
□ 9.3 圧縮ファイルを展開する	238
ONEPOINT■	圧縮ファイルを展開するには「Expand-Archive」を使用する

## CHAPTER □6 ドライブ・ロケーション

□ 9.4 PSプロバイダーを取得する	240
ONEPOINT■	PSプロバイダーを取得するには「Get-PSPProvider」を使用する
□ 9.5 PSドライブを取得する	241
ONEPOINT■	PSドライブを取得するには「Get-PSDrive」を使用する
□ 9.6 PSドライブを作成する	243
ONEPOINT■	PSドライブを作成するには「New-PSDrive」を使用する
COLUMN■	PowerShell外からでも使えるネットワークドライブを作成するには
□ 9.7 PSドライブを削除する	245
ONEPOINT■	PSドライブを削除するには「Remove-PSDrive」を使用する
□ 9.8 カレントディレクトリを移動する	246
ONEPOINT■	カレントディレクトリを移動するには「Set-Location」を使用する
□ 9.9 カレントディレクトリを取得する	247
ONEPOINT■	カレントディレクトリを取得するには「Get-Location」を使用する
COLUMN■	スタックに保存されたパスを表示するには

1 □ □ スタックに現在の場所を保存する .....	249
<b>ONEPOINT</b> ■ スタックに現在の場所を保存するには「Push-Location」を使用する	
<b>COLUMN</b> ■ 既定以外のスタックに保存するには	
1 □ 1 スタックから場所を取り出す .....	251
<b>ONEPOINT</b> ■ スタックから場所を取り出すには「Pop-Location」を使用する	
<b>COLUMN</b> ■ 既定以外のスタックから取り出した場所に変更するには	
1 □ 2 パスを変換する .....	252
<b>ONEPOINT</b> ■ パスを変換するには「Convert-Path」を使用する	
1 □ 3 パスを結合する .....	253
<b>ONEPOINT</b> ■ パスを結合するには「Join-Path」を使用する	
<b>COLUMN</b> ■ 結合されたパスによって参照される項目を表示するには	
1 □ 4 パスを解決する .....	255
<b>ONEPOINT</b> ■ パスを解決するには「Resolve-Path」を使用する	
<b>COLUMN</b> ■ 相対パスを取得するには	
1 □ 5 パスを分割する .....	257
<b>ONEPOINT</b> ■ パスを分割するには「Split-Path」を使用する	
<b>COLUMN</b> ■ 分割されたパスが参照している項目を表示するには	
1 □ 6 パスの存在をチェックする .....	259
<b>ONEPOINT</b> ■ パスの存在をチェックするには「Test-Path」を使用する	
<b>COLUMN</b> ■ パスの構文をチェックするには	

## CHAPTER 07 オブジェクト

1 □ 7 オブジェクトのメンバーを表示する .....	262
<b>ONEPOINT</b> ■ オブジェクトのメンバーを表示するには「Get-Member」を使用する	
<b>COLUMN</b> ■ コレクション自体のメンバーを取得するには	
<b>COLUMN</b> ■ オブジェクトの静的メンバーを取得するには	
<b>COLUMN</b> ■ 特定の種類のメンバーのみを取得するには	
1 □ 8 オブジェクトを既定の出力先に出力する .....	265
<b>ONEPOINT</b> ■ オブジェクトを既定の出力先に出力するには「Out-Default」を使用する	
1 □ 9 オブジェクトをコンソールに出力する .....	267
<b>ONEPOINT</b> ■ オブジェクトをコンソールに出力するには「Out-Host」を使用する	
1 □ 10 オブジェクトの出力を削除する .....	269
<b>ONEPOINT</b> ■ オブジェクトの出力を削除するには「Out-Null」を使用する	
1 □ 11 オブジェクトをプリンタに出力する .....	270
<b>ONEPOINT</b> ■ オブジェクトをプリンタに出力するには「Out-Printer」を使用する	
<b>COLUMN</b> ■ 代替プリンタで印刷するには	
1 □ 12 オブジェクトをファイルに出力する .....	272
<b>ONEPOINT</b> ■ オブジェクトをファイルに出力するには「Out-File」を使用する	
1 □ 13 オブジェクトを文字列として出力する .....	273
<b>ONEPOINT</b> ■ オブジェクトを文字列として出力するには「Out-String」を使用する	

114	オブジェクトをグリッドビューに表示する	275
ONEPOINT	■オブジェクトをグリッドビューに表示するには「Out-GridView」を使用する	
115	オブジェクトをリスト形式で出力する	277
ONEPOINT	■オブジェクトをリスト形式で出力するには「Format-List」を使用する	
COLUMN	■特定のプロパティを表示するには	
COLUMN	■プロパティをグループ化するには	
116	オブジェクトを表形式で出力する	279
ONEPOINT	■オブジェクトを表形式で出力するには「Format-Table」を使用する	
COLUMN	■特定のプロパティを表示するには	
117	オブジェクトを幅広い表形式で出力する	281
ONEPOINT	■オブジェクトを幅広い表形式で出力するには「Format-Wide」を使用する	
COLUMN	■列数を指定するには	
COLUMN	■特定のプロパティを表示するには	
118	オブジェクトをカスタムビューで出力する	283
ONEPOINT	■オブジェクトをカスタムビューで出力するには 「Format-Custom」を使用する	
COLUMN	■特定のプロパティを表示するには	
119	書式設定データを更新する	285
ONEPOINT	■書式設定データを更新するには「Update-FormatData」を使用する	
COLUMN	■書式設定ファイルについて	
120	書式設定データを取得する	287
ONEPOINT	■書式設定データを取得するには「Get-FormatData」を使用する	
COLUMN	■指定した型名の書式設定データを取得するには	
121	書式設定データを保存する	289
ONEPOINT	■書式設定データを保存するには「Export-FormatData」を使用する	
122	オブジェクトをCSV形式に変換する	291
ONEPOINT	■オブジェクトをCSV形式に変換するには「ConvertTo-CSV」を使用する	
COLUMN	■区切り文字を指定するには	
123	CSV形式からオブジェクトに変換する	293
ONEPOINT	■CSVからオブジェクトに変換するには「ConvertFrom-CSV」を使用する	
COLUMN	■列ヘッダーを変更するには	
124	オブジェクトをCSV形式ファイルで保存する	296
ONEPOINT	■オブジェクトをCSV形式ファイルで保存するには 「Export-CSV」を使用する	
125	CSV形式ファイルからオブジェクトに変換する	298
ONEPOINT	■CSV形式ファイルからオブジェクトに変換するには 「Import-CSV」を使用する	
126	オブジェクトをXMLに変換する	300
ONEPOINT	■オブジェクトをXMLに変換するには「ConvertTo-XML」を使用する	
COLUMN	■出力形式や階層数を指定するには	
127	オブジェクトをXMLファイルに保存する	302
ONEPOINT	■オブジェクトをXMLファイルに保存するには「Export-Clixml」を使用する	

1 2 8	XMLファイルからオブジェクトに変換する	304
ONEPOINT	■XMLファイルからオブジェクトに変換するには 「Import-Clixml」を使用する	
COLUMN	■取得数やスキップ数を指定するには	
1 2 9	オブジェクトをHTMLに変換する	306
ONEPOINT	■オブジェクトをHTMLに変換するには「ConvertTo-Html」を使用する	
COLUMN	■出力形式を指定するには	
1 3 0	オブジェクトをJSONに変換する	308
ONEPOINT	■オブジェクトをJSONに変換するには「ConvertTo-Json」を使用する	
COLUMN	■出力に含めるオブジェクトのレベル数を指定するには	
1 3 1	JSONからオブジェクトに変換する	310
ONEPOINT	■JSONからオブジェクトに変換するには 「ConvertFrom-Json」を使用する	
1 3 2	オブジェクトをランダムに選択する	312
ONEPOINT	■オブジェクトをランダムに選択するには「Get-Random」を使用する	
COLUMN	■ランダムな数値を取得するには	
1 3 3	重複しているオブジェクトを取り除く	314
ONEPOINT	■重複しているオブジェクトを取り除くには「Get-Unique」を使用する	
COLUMN	■「Get-Unique」の「InputObject」パラメータについて	
COLUMN	■データを文字列として扱うには	
COLUMN	■各型のオブジェクトを1つだけ取得するには	
1 3 4	オブジェクトを選択する	317
ONEPOINT	■オブジェクトを選択するには「Select-Object」を使用する	
COLUMN	■オブジェクトの配列から特定のオブジェクトを選択するには	
1 3 5	オブジェクトを並べ替える	320
ONEPOINT	■オブジェクトを並べ替えるには「Sort-Object」を使用する	
COLUMN	■並べ替え方法を指定するには	
1 3 6	オブジェクトをグループ化する	323
ONEPOINT	■オブジェクトをグループ化するには「Group-Object」を使用する	
COLUMN	■結果からグループのメンバーを除外するには	
COLUMN	■グループをハッシュテーブルとして取得するには	
1 3 7	オブジェクトをフィルターする	326
ONEPOINT	■オブジェクトをフィルターするには「Where-Object」を使用する	
COLUMN	■V3以降の追加構文でオブジェクトをフィルターするには	
1 3 8	オブジェクトをループ処理する	328
ONEPOINT	■オブジェクトをループ処理するには「ForEach-Object」を使用する	
COLUMN	■V3以降の追加構文でオブジェクトをループ処理するには	
1 3 9	オブジェクトを比較する	330
ONEPOINT	■オブジェクトを比較するには「Compare-Object」を使用する	
COLUMN	■すべての比較結果を表示するには	

1 4 □ オブジェクトを計測する	332
ONEPOINT■ オブジェクトを計測するには「Measure-Object」を使用する	
COLUMN■ 「最小値」「最大値」「合計」「平均」を求めるには	
COLUMN■ テキストファイルの「文字数」「行数」「語数」を計測するには	
1 4 1 オブジェクトを2方向に送信する	334
ONEPOINT■ オブジェクトを2方向に送信するには「Tee-Object」を使用する	
COLUMN■ オブジェクトを変数に送信するには	
1 4 2 オブジェクトを作成する	336
ONEPOINT■ オブジェクトを作成するには「New-Object」を使用する	
COLUMN■ COMオブジェクトを作成するには	
1 4 3 オブジェクトにメンバーを追加する	338
ONEPOINT■ オブジェクトにメンバーを追加するには「Add-Member」を使用する	
COLUMN■ V3以降の追加構文でノートプロパティを追加するには	
1 4 4 クラスを追加する	340
ONEPOINT■ クラスを追加するには「Add-Type」を使用する	
COLUMN■ ソースコードをファイルから読み込むには	
COLUMN■ PowerShellでネイティブWindows APIを呼び出すには	
1 4 5 型データを取得する	343
ONEPOINT■ 型データを取得するには「Get-TypeData」を使用する	
1 4 6 型データを更新する	344
ONEPOINT■ 型データを更新するには「Update-TypeData」を使用する	
COLUMN■ 型定義ファイルについて	
COLUMN■ コマンド入力で型を拡張するには	
1 4 7 型データを削除する	347
ONEPOINT■ 型データを削除するには「Remove-TypeData」を使用する	
COLUMN■ 型データを指定して型データを削除するには	
COLUMN■ 型定義ファイルに定義されている型データを削除するには	

## CHAPTER 08 スナップイン・モジュール

1 4 8 スナップインを追加する	350
ONEPOINT■ スナップインを追加するには「Add-PSSnapin」を使用する	
1 4 9 スナップインを取得する	351
ONEPOINT■ スナップインを取得するには「Get-PSSnapin」を使用する	
1 5 □ スナップインを削除する	352
ONEPOINT■ スナップインを削除するには「Remove-PSSnapin」を使用する	
1 5 1 スナップインの情報をファイルに保存する	353
ONEPOINT■ スナップインの情報をファイルに保存するには 「Export-Console」を使用する	
COLUMN■ 保存したコンソールファイルを新しいセッションで使用するには	

152 モジュールを追加する .....	355
<b>ONEPOINT</b> ■ モジュールを追加するには「Import-Module」を使用する	
<b>COLUMN</b> ■ モジュールについて	
<b>COLUMN</b> ■ モジュールのメンバーを取得するには	
<b>COLUMN</b> ■ 指定したメンバーのみをインポートするには	
<b>COLUMN</b> ■ プレフィックスを追加するには	
153 モジュールを取得する .....	359
<b>ONEPOINT</b> ■ モジュールを取得するには「Get-Module」を使用する	
<b>COLUMN</b> ■ 特定のモジュールを取得するには	
<b>COLUMN</b> ■ 利用可能なモジュールをすべてインポートするには	
154 モジュールを削除する .....	362
<b>ONEPOINT</b> ■ モジュールを削除するには「Remove-Module」を使用する	
155 動的モジュールを作成する .....	363
<b>ONEPOINT</b> ■ 動的モジュールを作成するには「New-Module」を使用する	
<b>COLUMN</b> ■ 動的モジュールを取得するには	
<b>COLUMN</b> ■ 指定したメンバーのみをエクスポートするには	
<b>COLUMN</b> ■ カスタムオブジェクトを取得するには	
156 モジュールのメンバーをエクスポートする .....	366
<b>ONEPOINT</b> ■ モジュールのメンバーをエクスポートするには 「Export-ModuleMember」を使用する	
157 モジュールマニフェストを作成する .....	367
<b>ONEPOINT</b> ■ モジュールマニフェストを作成するには 「New-ModuleManifest」を使用する	
<b>COLUMN</b> ■ マニフェストキーの値を設定するには	
158 モジュールマニフェストをテストする .....	369
<b>ONEPOINT</b> ■ モジュールマニフェストをテストするには 「Test-ModuleManifest」を使用する	

## CHAPTER 09 システム・プロセス・サービス

159 ホスト情報を取得する .....	372
<b>ONEPOINT</b> ■ ホスト情報を取得するには「Get-Host」を使用する	
<b>COLUMN</b> ■ PowerShellコンソールのUIのカスタマイズするには	
160 地域設定に関する情報を取得する .....	374
<b>ONEPOINT</b> ■ カルチャは「Get-Culture」、UIカルチャは「Get-UICulture」で取得する	
<b>COLUMN</b> ■ カルチャのすべてのプロパティを取得するには	
161 イベントログを作成する .....	376
<b>ONEPOINT</b> ■ イベントログを作成するには「New-EventLog」を使用する	
162 イベントをイベントログに書き込む .....	378
<b>ONEPOINT</b> ■ イベントをイベントログに書き込むには「Write-EventLog」を使用する	
163 イベント/イベントログを取得する .....	380
<b>ONEPOINT</b> ■ イベント/イベントログを取得するには「Get-EventLog」を使用する	

164	イベントビューアーを起動する	382
	<b>ONEPOINT</b> ■ イベントビューアーを起動するには「Show-EventLog」を使用する	
165	イベントログを制限する	384
	<b>ONEPOINT</b> ■ イベントログを制限するには「Limit-EventLog」を使用する	
166	イベントログのエントリを削除する	385
	<b>ONEPOINT</b> ■ イベントログのエントリを削除するには「Clear-EventLog」を使用する	
167	イベントログを削除する	387
	<b>ONEPOINT</b> ■ イベントログを削除するには「Remove-EventLog」を使用する	
168	新形式のイベントログを取得する	389
	<b>ONEPOINT</b> ■ 新形式のイベントログを取得するには「Get-WinEvent」を使用する	
	<b>COLUMN</b> ■ 新形式のイベントログについて	
	<b>COLUMN</b> ■ イベントログプロバイダーの一覧を取得するには	
	<b>COLUMN</b> ■ イベントをフィルター処理するには	
169	新形式のイベントを作成する	392
	<b>ONEPOINT</b> ■ 新形式のイベントを作成するには「New-WinEvent」を使用する	
170	復元ポイントを作成する	393
	<b>ONEPOINT</b> ■ 復元ポイントを作成するには「Checkpoint-Computer」を使用する	
171	復元ポイントを取得する	395
	<b>ONEPOINT</b> ■ 復元ポイントを取得するには 「Get-ComputerRestorePoint」を使用する	
	<b>COLUMN</b> ■ 特定のシーケンス番号の復元ポイントを取得するには	
	<b>COLUMN</b> ■ 最近試みた復元の状態を取得するには	
172	システム復元機能を無効にする	397
	<b>ONEPOINT</b> ■ システム復元機能を無効にするには 「Disable-ComputerRestore」を使用する	
173	システム復元機能を有効にする	399
	<b>ONEPOINT</b> ■ システム復元機能を有効にするには 「Enable-ComputerRestore」を使用する	
174	システムの復元を実行する	401
	<b>ONEPOINT</b> ■ システムの復元を実行する「Restore-Computer」を使用する	
175	修正プログラムを取得する	402
	<b>ONEPOINT</b> ■ 修正プログラムを取得するには「Get-HotFix」を使用する	
	<b>COLUMN</b> ■ 特定の修正プログラムを取得するには	
176	コンピュータを再起動する	404
	<b>ONEPOINT</b> ■ コンピュータを再起動するには「Restart-Computer」を使用する	
177	コンピュータをシャットダウンする	405
	<b>ONEPOINT</b> ■ コンピュータをシャットダウンするには「Stop-Computer」を使用する	
178	プロセスを開始する	406
	<b>ONEPOINT</b> ■ プロセスを開始するには「Start-Process」を使用する	
	<b>COLUMN</b> ■ プロセスに使用するウィンドウの状態を指定するには	
	<b>COLUMN</b> ■ プロセスを開始するときの動作を指定するには	

179	プロセスを停止する	408
	<b>ONEPOINT</b> ■ プロセスを停止するには「Stop-Process」を使用する	
	<b>COLUMN</b> ■ プロセスの特定のインスタンスを停止するには	
180	プロセスを取得する	410
	<b>ONEPOINT</b> ■ プロセスを取得するには「Get-Process」を使用する	
181	プロセスが停止するまで待機する	412
	<b>ONEPOINT</b> ■ プロセスが停止するまで待機するには「Wait-Process」を使用する	
182	プロセスをデバッグする	413
	<b>ONEPOINT</b> ■ プロセスをデバッグするには「Debug-Process」を使用する	
183	サービスを取得する	414
	<b>ONEPOINT</b> ■ サービスを取得するには「Get-Service」を使用する	
184	サービスを開始する	416
	<b>ONEPOINT</b> ■ サービスを開始するには「Start-Service」を使用する	
	<b>COLUMN</b> ■ サービスが開始可能かどうかを調べるには	
185	サービスを停止する	418
	<b>ONEPOINT</b> ■ サービスを停止するには「Stop-Service」を使用する	
186	サービスを中断する	420
	<b>ONEPOINT</b> ■ サービスを中断するには「Suspend-Service」を使用する	
	<b>COLUMN</b> ■ サービスが中断可能かどうかを確認するには	
187	サービスを再開する	422
	<b>ONEPOINT</b> ■ サービスを再開するには「Resume-Service」を使用する	
188	サービスを再起動する	424
	<b>ONEPOINT</b> ■ サービスを再起動するには「Restart-Service」を使用する	
189	サービスを設定する	425
	<b>ONEPOINT</b> ■ サービスを設定するには「Set-Service」を使用する	
	<b>COLUMN</b> ■ リモートコンピュータのサービスを設定するには	
190	サービスを新規作成する	427
	<b>ONEPOINT</b> ■ サービスを新規作成するには「New-Service」を使用する	
191	パフォーマンスカウンターデータを取得する	429
	<b>ONEPOINT</b> ■ パフォーマンスカウンターデータを取得するには 「Get-Counter」を使用する	
	<b>COLUMN</b> ■ カウンターパスの形式	
	<b>COLUMN</b> ■ パフォーマンスカウンターを継続的に取得するには	
	<b>COLUMN</b> ■ カウンターセットを取得するには	
192	パフォーマンスカウンターデータを記録する	432
	<b>ONEPOINT</b> ■ パフォーマンスカウンターデータを記録するには 「Export-Counter」を使用する	
	<b>COLUMN</b> ■ 出力形式を指定するには	
	<b>COLUMN</b> ■ 循環ログを作成するには	

193	カウンターデータを読み込む	434
ONEPOINT	■カウンターデータを読み込むには「Import-Counter」を使用する	
COLUMN	■指定したカウンターデータのみをインポートするには	
COLUMN	■インポートするサンプルの最大数を指定するには	
194	コントロールパネル項目を取得する	437
ONEPOINT	■コントロールパネル項目を取得するには 「Get-ControlPanelItem」を使用する	
COLUMN	■指定されたカテゴリ内のコントロールパネル項目を取得するには	
195	コントロールパネル項目を表示する	439
ONEPOINT	■コントロールパネル項目を表示するには 「Show-ControlPanelItem」を使用する	

## CHAPTER 10 セキュリティ・ネットワーク

196	実行ポリシーを変更する	442
ONEPOINT	■実行ポリシーを変更するには「Set-ExecutionPolicy」を使用する	
COLUMN	■特定のスコープに実行ポリシーを設定するには	
197	実行ポリシーを取得する	444
ONEPOINT	■実行ポリシーを取得するには「Get-ExecutionPolicy」を使用する	
198	ACLを取得する	445
ONEPOINT	■ACLを取得するには「Get-Acl」を使用する	
199	ACLを変更する	447
ONEPOINT	■ACLを変更するには「Set-Acl」を使用する	
200	スクリプトファイルに署名する	449
ONEPOINT	■スクリプトファイルに署名するには 「Set-AuthenticodeSignature」を使用する	
COLUMN	■証明書を作成するには	
201	スクリプトファイルの署名を取得する	453
ONEPOINT	■スクリプトファイルの署名を取得するには 「Get-AuthenticodeSignature」を使用する	
202	pfx証明書ファイルの情報を取得する	454
ONEPOINT	■pfx証明書ファイルの情報を取得するには 「Get-PfxCertificate」を使用する	
COLUMN	■pfx証明書ファイルをエクスポートするには	
203	ファイルのハッシュ値を取得する	456
ONEPOINT	■ファイルのハッシュ値を取得するには「Get-FileHash」を使用する	
204	資格情報オブジェクトを作成する	457
ONEPOINT	■資格情報オブジェクトを作成するには「Get-Credential」を使用する	
205	コンピュータ名を変更する	459
ONEPOINT	■コンピュータ名を変更するには「Rename-Computer」を使用する	
COLUMN	■リモートコンピュータ名を変更するには	

206	コンピュータをワークグループ/ドメインに追加する	461
<b>ONEPOINT</b> ■コンピュータをワークグループ/ドメインに追加するには 「Add-Computer」を使用する		
<b>COLUMN</b> ■コンピュータをドメインに追加するには		
207	ドメインからコンピュータを削除する	463
<b>ONEPOINT</b> ■ドメインからコンピュータを削除するには 「Remove-Computer」を使用する		
208	pingを送信する	464
<b>ONEPOINT</b> ■pingを送信するには「Test-Connection」を使用する		
209	ドメイン認証用のパスワードをリセットする	466
<b>ONEPOINT</b> ■ドメイン認証用のパスワードをリセットするには 「Reset-ComputerMachinePassword」を使用する		
210	セキュアチャンネルをテストする	467
<b>ONEPOINT</b> ■セキュアチャンネルをテストするには 「Test-ComputerSecureChannel」を使用する		
211	メールを送信する	468
<b>ONEPOINT</b> ■メールを送信するには「Send-MailMessage」を使用する		
212	CMS形式でコンテンツを暗号化する	469
<b>ONEPOINT</b> ■CMS形式でコンテンツを暗号化するには 「Protect-CmsMessage」を使用する		
<b>COLUMN</b> ■暗号化されたコンテンツをファイルに保存するには		
213	CMS形式で暗号化されたコンテンツを取得する	472
<b>ONEPOINT</b> ■CMS形式で暗号化されたコンテンツを取得するには 「Get-CmsMessage」を使用する		
214	CMS形式暗号化コンテンツを復号化する	474
<b>ONEPOINT</b> ■CMS形式暗号化コンテンツを復号化するには 「UnProtect-CmsMessage」を使用する		
<b>COLUMN</b> ■パイプ処理で暗号化コンテンツを復号化するには		
215	ファイルのブロックを解除する	476
<b>ONEPOINT</b> ■ファイルのブロックを解除するには「Unblock-File」を使用する		
<b>COLUMN</b> ■ファイルのブロックについて		
<b>COLUMN</b> ■ブロックされているファイルを取得するには		
216	Webサービスプロキシオブジェクトを作成する	478
<b>ONEPOINT</b> ■Webサービスプロキシオブジェクトを作成するには 「New-WebServiceProxy」を使用する		
<b>COLUMN</b> ■Webサービスプロキシオブジェクトのメソッドを取得するには		
<b>COLUMN</b> ■名前空間とクラス名を指定するには		
217	Webページのコンテンツを取得する	481
<b>ONEPOINT</b> ■Webページのコンテンツを取得するには 「Invoke-WebRequest」を使用する		
<b>COLUMN</b> ■HTML要素のコレクションを取得するには		
<b>COLUMN</b> ■ファイルをダウンロードするには		
<b>COLUMN</b> ■フォームの入力データを送信するには		

218	Webサービスからデータを取得する	484
ONEPOINT	■Webサービスからデータを取得するには 「Invoke-WebRequest」を使用する	
COLUMN	■「Invoke-WebRequest」の出力形式について	

## CHAPTER 11 ジョブ・ワークフロー

219	ジョブを開始する	488
ONEPOINT	■ジョブを開始するには「Start-Job」を使用する	
COLUMN	■スクリプトファイルをバックグラウンドジョブとして実行するには	
COLUMN	■フレンドリ名を指定するには	
COLUMN	■コマンドにオブジェクトを入力するには	
220	ジョブを取得する	491
ONEPOINT	■ジョブを取得するには「Get-Job」を使用する	
COLUMN	■特定のジョブを取得するには	
COLUMN	■「Filter」パラメータの指定方法	
221	ジョブが完了するまで待機する	494
ONEPOINT	■ジョブが完了するまで待機するには「Wait-Job」を使用する	
COLUMN	■いずれかのジョブの完了を待機するには	
222	ジョブを停止する	496
ONEPOINT	■ジョブを停止するには「Stop-Job」を使用する	
223	ジョブの結果を取得する	498
ONEPOINT	■ジョブの結果を取得するには「Receive-Job」を使用する	
224	ジョブを削除する	500
ONEPOINT	■ジョブを削除するには「Remove-Job」を使用する	
225	ワークフロージョブを一時停止する	502
ONEPOINT	■ワークフロージョブを一時停止するには「Suspend-Job」を使用する	
226	ワークフロージョブを再開する	504
ONEPOINT	■ワークフロージョブを再開するには「Resume-Job」を使用する	
227	ワークフローセッションを作成する	506
ONEPOINT	■ワークフローセッションを作成するには 「New-PSWorkflowSession」を使用する	
228	ワークフローセッション構成オプションを作成する	508
ONEPOINT	■ワークフローセッション構成オプションを作成するには 「New-PSWorkflowExecutionOption」を使用する	
229	コマンドをワークフローとして実行する	510
ONEPOINT	■コマンドをワークフローとして実行するには 「Invoke-AsWorkflow」を使用する	
COLUMN	■式をワークフローとして実行するには	

## CHAPTER 12 リモート・セッション

230	PSリモート機能を有効にする	514
ONEPOINT ■ PSリモート機能を有効にするには「Enable-PSRemoting」を使用する		
COLUMN ■ ホストコンピュータを信頼できるホストとして登録する		
COLUMN ■ パブリックネットワークでPSリモート機能を有効にするには		
231	PSリモート機能を無効にする	516
ONEPOINT ■ PSリモート機能を無効にするには「Disable-PSRemoting」を使用する		
232	PSセッション構成を有効にする	517
ONEPOINT ■ PSセッション構成を有効にするには「Enable-PSSessionConfiguration」を使用する		
COLUMN ■ 指定したPSセッション構成を有効にするには		
COLUMN ■ セキュリティ記述を置換するには		
233	PSセッション構成を取得する	519
ONEPOINT ■ PSセッション構成を取得するには 「Get-PSSessionConfiguration」を使用する		
234	PSセッション構成を無効にする	521
ONEPOINT ■ PSセッション構成を無効にするには「Disable-PSSessionConfiguration」を使用する		
COLUMN ■ 指定したPSセッション構成を無効にするには		
235	PSセッション構成を登録する	523
ONEPOINT ■ PSセッション構成を登録するには「Register-PSSessionConfiguration」を使用する		
COLUMN ■ セッション構成について		
236	PSセッション構成を変更する	527
ONEPOINT ■ PSセッション構成を変更するには 「Set-PSSessionConfiguration」を使用する		
237	PSセッション構成を削除する	530
ONEPOINT ■ PSセッション構成を削除するには 「Unregister-PSSessionConfiguration」を使用する		
238	転送オプションを作成する	532
ONEPOINT ■ 転送オプションを作成するには「New-PSTransportOption」を使用する		
239	PSセッション構成ファイルを作成する	534
ONEPOINT ■ PSセッション構成ファイルを作成するには 「New-PSSessionConfigurationFile」を使用する		
COLUMN ■ セッション構成ファイルについて		
240	PSセッション構成ファイルをテストする	538
ONEPOINT ■ PSセッション構成ファイルをテストするには 「Test-PSSessionConfigurationFile」を使用する		
241	PSセッションを作成する	539
ONEPOINT ■ PSセッションを作成するには「New-PSSession」を使用する		
COLUMN ■ PSセッションの作成方法		

<b>242</b>	<b>PSセッションを取得する</b>	541
<b>ONEPOINT</b>	■ PSセッションを取得するには「Get-PSSession」を使用する	
<b>COLUMN</b>	■ 特定のPSセッションを取得する	
<b>COLUMN</b>	■ 特定のコンピュータに接続しているPSセッションを取得するには	
<b>243</b>	<b>PSセッションを切断する</b>	544
<b>ONEPOINT</b>	■ PSセッションを切断するには「Disconnect-PSSession」を使用する	
<b>244</b>	<b>PSセッションに接続する</b>	546
<b>ONEPOINT</b>	■ PSセッションに接続するには「Connect-PSSession」を使用する	
<b>COLUMN</b>	■ 特定のコンピュータに接続しているPSセッションに接続するには	
<b>245</b>	<b>対話型セッションを開始する</b>	549
<b>ONEPOINT</b>	■ 対話型セッションを開始するには「Enter-PSSession」を使用する	
<b>COLUMN</b>	■ 作成済みのPSセッションで対話型セッションを開始するには	
<b>246</b>	<b>対話型セッションを終了する</b>	552
<b>ONEPOINT</b>	■ 対話型セッションを終了するには「Exit-PSSession」を使用する	
<b>247</b>	<b>PSセッションオプションを作成する</b>	553
<b>ONEPOINT</b>	■ PSセッションオプションを作成するには 「New-PSSessionOption」を使用する	
<b>248</b>	<b>PSセッションの結果を取得する</b>	555
<b>ONEPOINT</b>	■ PSセッションの結果を取得するには「Receive-PSSession」を使用する	
<b>COLUMN</b>	■ 特定のコンピュータの切断セッションを指定するには	
<b>249</b>	<b>PSセッションを削除する</b>	558
<b>ONEPOINT</b>	■ PSセッションを削除するには「Remove-PSSession」を使用する	
<b>COLUMN</b>	■ 「Remove-PSSession」の実行結果について	
<b>250</b>	<b>PSセッションのコマンドをインポートする</b>	560
<b>ONEPOINT</b>	■ PSセッションのコマンドをインポートするには 「Import-PSSession」を使用する	
<b>COLUMN</b>	■ インポートされたコマンドをバックグラウンドジョブとして実行するには	
<b>COLUMN</b>	■ モジュールをインポートするには	
<b>COLUMN</b>	■ 書式設定データを含めてインポートするには	
<b>251</b>	<b>PSセッションのコマンドをエクスポートする</b>	563
<b>ONEPOINT</b>	■ PSセッションのコマンドをエクスポートするには 「Export-PSSession」を使用する	
<b>COLUMN</b>	■ エクスポートされたモジュールをインポートするには	
<b>COLUMN</b>	■ 書式設定データを含めてエクスポートするには	
<b>252</b>	<b>リモートコンピュータでコマンドを実行する</b>	565
<b>ONEPOINT</b>	■ リモートコンピュータでコマンドを実行するには 「Invoke-Command」を使用する	
<b>COLUMN</b>	■ 関連するコマンドを続けて実行するには	
<b>COLUMN</b>	■ コマンドにオブジェクトを入力するには	

## CHAPTER 13 WSMAN・WMI

253	WinRMサービスへの接続を作成する	570
ONEPOINT ■ WinRMサービスへの接続を作成するには 「Connect-WSMan」を使用する		
254	WinRMサービスへの接続を削除する	572
ONEPOINT ■ WinRMサービスへの接続を削除するには 「Disconnect-WSMan」を使用する		
255	WSManインスタンスを取得する	573
ONEPOINT ■ WSMANインスタンスを取得するには 「Get-WSManInstance」を使用する		
COLUMN ■ WSMANインスタンス内の特定のセクションを取得するには		
COLUMN ■ すべてのWSMANインスタンスを取得するには		
256	WSManインスタンスを作成する	575
ONEPOINT ■ WSMANインスタンスを作成するには 「New-WSManInstance」を使用する		
COLUMN ■ HTTPSリスナー用の証明書を作成するには		
257	WSManインスタンスを削除する	578
ONEPOINT ■ WSMANインスタンスを削除するには 「Remove-WSManInstance」を使用する		
258	WSManインスタンスを設定する	580
ONEPOINT ■ WSMANインスタンスを設定するには 「Set-WSManInstance」を使用する		
259	WSManインスタンスのメソッドを実行する	582
ONEPOINT ■ WSMANインスタンスのメソッドを実行するには 「Invoke-WSManAction」を使用する		
COLUMN ■ WSMANインスタンスのメソッドにパラメータ値を渡すには		
260	WSManセッションオプションを作成する	584
ONEPOINT ■ WSMANセッションオプションを作成するには 「New-WSManSessionOption」を使用する		
261	CredSSP認証を有効にする	586
ONEPOINT ■ CredSSP認証を有効にするには 「Enable-WSManCredSSP」を使用する		
COLUMN ■ 「Role」パラメータの設定値と実行される処理		
262	CredSSPの構成情報を取得する	588
ONEPOINT ■ CredSSPの構成情報を取得するには 「Get-WSManCredSSP」を使用する		
263	CredSSP認証を無効にする	590
ONEPOINT ■ CredSSP認証を無効にするには 「Disable-WSManCredSSP」を使用する		
264	WinRMリモート管理を有効にする	591
ONEPOINT ■ WinRMリモート管理を有効にするには 「Set-WSManQuickConfig」を使用する		

265	WinRMサービスの実行を確認する	592
<b>ONEPOINT</b> ■WinRMサービスの実行を確認するには「Test-WSMan」を使用する		
266	WMIオブジェクトを取得する	593
<b>ONEPOINT</b> ■WMIオブジェクトを取得するには「Get-WmiObject」を使用する		
<b>COLUMN</b> ■指定したWMIオブジェクトを取得するには		
<b>COLUMN</b> ■指定したプロパティを取得するには		
<b>COLUMN</b> ■WMIオブジェクトのメソッドを実行するには		
<b>COLUMN</b> ■指定した名前空間内にあるWMIクラスを取得するには		
267	WMIインスタンスを設定する	596
<b>ONEPOINT</b> ■WMIインスタンスを設定するには「Set-WmiInstance」を使用する		
268	WMIオブジェクトを削除する	598
<b>ONEPOINT</b> ■WMIオブジェクトを削除するには「Remove-WmiObject」を使用する		
269	WMIメソッドを実行する	600
<b>ONEPOINT</b> ■WMIメソッドを実行するには「Invoke-WmiMethod」を使用する		

## CHAPTER 14 イベント

270	カスタムイベントを作成する	604
<b>ONEPOINT</b> ■カスタムイベントを作成するには「New-Event」を使用する		
<b>COLUMN</b> ■カスタムイベントについて		
271	イベントを取得する	606
<b>ONEPOINT</b> ■イベントを取得するには「Get-Event」を使用する		
<b>COLUMN</b> ■イベントキューについて		
272	イベントの発生を待つ	608
<b>ONEPOINT</b> ■イベントの発生を待つには「Wait-Event」を使用する		
<b>COLUMN</b> ■イベントの発生を待つ最大時間を指定するには		
273	イベントを削除する	610
<b>ONEPOINT</b> ■イベントを削除するには「Remove-Event」を使用する		
274	PowerShellエンジンのイベントを登録する	612
<b>ONEPOINT</b> ■PowerShellエンジンのイベントを登録するには 「Register-EngineEvent」を使用する		
<b>COLUMN</b> ■イベントサブスクリーバーについて		
275	.NETオブジェクトのイベントを登録する	614
<b>ONEPOINT</b> ■.NETオブジェクトのイベントを登録するには 「Register-ObjectEvent」を使用する		
<b>COLUMN</b> ■イベントの名前を検索するには		
276	WMIイベントを登録する	616
<b>ONEPOINT</b> ■WMIイベントを登録するには「Register-WmiEvent」を使用する		
277	イベントサブスクリーバーを取得する	618
<b>ONEPOINT</b> ■イベントサブスクリーバーを取得するには 「Get-EventSubscriber」を使用する		

278	イベントサブスクリプションをキャンセルする	620
ONEPOINT	■イベントサブスクリプションをキャンセルするには 「Unregister-Event」を使用する	
COLUMN	■すべてのイベントサブスクリプションをキャンセルするには	

## CHAPTER 15 デバッグ

279	セッションを記録する	624
ONEPOINT	■セッションを記録するには「Start-Transcript」を使用する	
COLUMN	■出力ファイルの保存先を指定するには	
280	スクリプトやセッションを中断する	626
ONEPOINT	■スクリプトやセッションを中断するには「Start-Sleep」を使用する	
281	トレースソースを取得する	627
ONEPOINT	■トレースソースを取得するには「Get-TraceSource」を使用する	
282	コンポーネントをトレースする	628
ONEPOINT	■コンポーネントをトレースするには「Set-TraceSource」を使用する	
COLUMN	■ファイルリスナーを使用するトレースを停止するには	
283	コマンドをトレースする	630
ONEPOINT	■コマンドをトレースするには「Trace-Command」を使用する	
COLUMN	■コマンドにオブジェクトを入力するには	
284	Strictモードを設定する	632
ONEPOINT	■Strictモードを設定するには「Set-StrictMode」を使用する	
COLUMN	■Strictモードについて	
285	デバッグ機能を設定する	634
ONEPOINT	■デバッグ機能を設定するには「Set-PSDebug」を使用する	
COLUMN	■ステップ処理を有効にするには	
COLUMN	■Strictモードを有効にするには	
286	ブレークポイントを設定する	636
ONEPOINT	■ブレークポイントを設定するには「Set-PSBreakpoint」を使用する	
COLUMN	■コマンドブレークポイントを設定するには	
COLUMN	■変数ブレークポイントを設定するには	
COLUMN	■条件付きブレークポイントを設定するには	
COLUMN	■デバッカーコマンドについて	
287	ブレークポイントを取得する	640
ONEPOINT	■ブレークポイントを取得するには「Get-PSBreakpoint」を使用する	
288	ブレークポイントを無効にする	642
ONEPOINT	■ブレークポイントを無効にするには「Disable-PSBreakpoint」を使用する	
289	ブレークポイントを有効にする	644
ONEPOINT	■ブレークポイントを有効にするには「Enable-PSBreakpoint」を使用する	
290	ブレークポイントを削除する	646
ONEPOINT	■ブレークポイントを削除するには「Remove-PSBreakpoint」を使用する	

291	デバッガーでスクリプトを停止する	647
	<b>ONEPOINT</b> ■デバッガーでスクリプトを停止するには「Wait-Debugger」を使用する	
292	ジョブをデバッグする	648
	<b>ONEPOINT</b> ■ジョブをデバッグするには「Debug-Job」を使用する	
293	コールスタックを取得する	650
	<b>ONEPOINT</b> ■コールスタックを取得するには「Get-PSCallStack」を使用する	
294	ホストプロセスに接続する	652
	<b>ONEPOINT</b> ■ホストプロセスに接続するには「Enter-PSHostProcess」を使用する	
295	ホストプロセスから切断する	654
	<b>ONEPOINT</b> ■ホストプロセスから切断するには「Exit-PSHostProcess」を使用する	
296	ホストプロセスの実行空間を取得する	655
	<b>ONEPOINT</b> ■ホストプロセスの実行空間を取得するには「Get-Runspace」を使用する	
297	実行空間をデバッグする	657
	<b>ONEPOINT</b> ■実行空間をデバッグするには「Debug-Runspace」を使用する	
	<b>COLUMN</b> ■リモートの実行空間をデバッグするには	
298	実行空間のデバッグオプションを取得する	659
	<b>ONEPOINT</b> ■実行空間のデバッグオプションを取得するには 「Get-RunspaceDebug」を使用する	
299	実行空間のデバッグを有効にする	661
	<b>ONEPOINT</b> ■実行空間のデバッグを有効にするには 「Enable-RunspaceDebug」を使用する	
300	実行空間のデバッグを無効にする	663
	<b>ONEPOINT</b> ■実行空間のデバッグを無効にするには 「Disable-RunspaceDebug」を使用する	

## CHAPTER 16 トランザクション

301	トランザクションを開始する	666
	<b>ONEPOINT</b> ■トランザクションを開始するには「Start-Transaction」を使用する	
	<b>COLUMN</b> ■ロールバック設定を変更するには	
	<b>COLUMN</b> ■タイムアウト時間を指定するには	
	<b>COLUMN</b> ■トランザクション進行中に別のトランザクションを開始するには	
302	トランザクションを取得する	670
	<b>ONEPOINT</b> ■トランザクションを取得するには「Get-Transaction」を使用する	
	<b>COLUMN</b> ■トランザクションをサポートするコマンドレットを検索するには	
303	トランザクションを取り消す	672
	<b>ONEPOINT</b> ■トランザクションを取り消すには「Undo-Transaction」を使用する	
304	トランザクションを完了する	674
	<b>ONEPOINT</b> ■トランザクションを完了するには「Complete-Transaction」を使用する	

305 トランザクションにスクリプトを追加する .....	676
ONEPOINT ■トランザクションにスクリプトを追加するには 「Use-Transaction」を使用する	
COLUMN ■トランザクション対応のプロバイダーを検索するには	

## CHAPTER 17 コア関数、その他

306 コンソールをクリアする .....	680
ONEPOINT ■コンソールをクリアするには「Clear-Host」を使用する	
307 コマンドレットで承認されている動詞を取得する .....	681
ONEPOINT ■コマンドレットで承認されている動詞を取得するには 「Get-Verb」を使用する	
308 ネットワークの場所を変更する .....	682
ONEPOINT ■ネットワークの場所を変更するには 「Set-NetConnectionProfile」を使用する	
309 文字列を処理する .....	683
ONEPOINT ■文字列を処理するには文字列オブジェクトのメンバーを使用する	
COLUMN ■文字列オブジェクトの特定のメンバーの構文を取得するには	
310 日付時刻を処理する .....	687
ONEPOINT ■日付時刻を処理するには日付時刻オブジェクトのメンバーを使用する	
COLUMN ■日付時刻オブジェクトの特定のメンバーの構文を取得するには	
311 ハッシュテーブルを処理する .....	690
ONEPOINT ■ハッシュテーブルを処理するには ハッシュテーブルオブジェクトのメンバーを使用する	
COLUMN ■ハッシュテーブルオブジェクトの特定のメンバーの構文を取得するには	
312 数学関数を使用する .....	693
ONEPOINT ■数学関数を使用するには 「math」クラスのスタティックメンバーを使用する	
COLUMN ■「math」クラスの特定のスタティックメンバーの構文を取得するには	
313 出力を整形する .....	695
ONEPOINT ■出力を整形するにはフォーマット演算子を使用する	
COLUMN ■コマンドレットの出力を整形するには	

## APPENDIX パラメータの詳細

314 パラメータの指定方法 .....	698
315 パラメータの有効値 .....	700
316 スイッチパラメータ .....	719
317 「PassThru」パラメータ .....	723

●索引 .....	725
-----------	-----

CHAPTER  1

---

# PowerShellの 基礎知識

# PowerShellとは

## III PowerShellの概要

PowerShell(正式には「Windows PowerShell」)とは、Microsoftが提供するシステム管理用のコマンドラインシェル(対話的にOSを操作するためのソフトウェア)およびスクリプト言語です。「コマンドレット」と呼ばれるコマンド(単機能の命令)を使用して対話的に作業を行うことができます。また、コマンドレットを組み合わせて、複雑な処理をスクリプトとして記述して実行することができます。

### ▶ 標準装備だから気軽に始められる

PowerShellは、Windows Server 2008 R2、Windows 7から標準装備されているので、気軽に始めることができます。それ以前のバージョンでは、インストールすることで利用できます(32ページ参照)。

### ▶ 一貫したコマンドレット体型

PowerShellのコマンドレットは、標準的な名前付けパターン([動詞]-[名詞])を使用しているので、コマンドレットの動作を容易に推測できます。また、コマンドレットには、豊富なオプションが容易されおり、オプションパラメータを指定することで、コマンドレットの動作を変更することができます(39ページ参照)。

### ▶ .NET Frameworkとの連携

PowerShellは、CLR(.NET Frameworkアプリケーションを実行するための仮想マシン)上で動作します。また、コマンドレットの戻り値(結果)は常に「オブジェクト」で、.NET Frameworkのクラスを呼び出すことができます。

### ▶ データへのアクセスの共通化

PowerShellでは、ファイルシステム、レジストリ、デジタル署名、環境変数、エイリアス、スクリプト変数、関数、WS-Management構成情報(2.0以降)をドライブ(PSドライブ)として扱うことができます。これらのドライブには、ドライブレターだけでなく、文字列を使用してアクセスすることができます(53ページ参照)。

### ▶ 統合スクリプティング開発環境によるスクリプトの作成

PowerShell 2.0以降には、Windows PowerShell Integrated Scripting Environment(以降「ISE」)が標準装備されています。ISEは、スクリプトおよびモジュールを記述、実行、およびテストすることのできる統合開発環境です(31ページ参照)。

### III PowerShellコンソール/ISEについて

PowerShellには、コンソール版(PowerShellコンソール)とGUI版(ISE)があります。

#### ▶ コンソール版

コンソール版のPowerShellは、従来のコマンドプロンプトと同様のコマンドラインのシェルです。「PowerShellコンソール」と呼ぶこともあります。キーボードからコマンドを入力して、**Enter**キーを押すことで、結果を得ることができます。

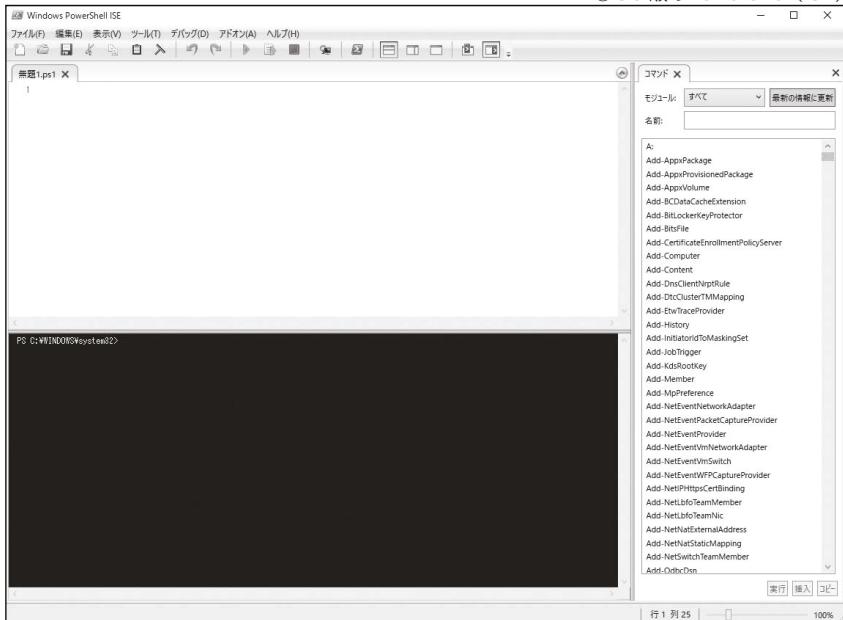
◎ コンソール版のPowerShell



#### ▶ GUI版(ISE)

GUI版のPowerShellは、「ISE」(Windows PowerShell Integrated Scripting Environment)と呼ばれ、PowerShell2.0から搭載された統合開発環境です。スクリプトおよびモジュールを記述、実行、およびテスト(デバッグ)を行うことができます。

◎ GUI版のPowerShell(ISE)



□ 1	CHAPTER
□ 2	
□ 3	
□ 4	
□ 5	
□ 6	
□ 7	
□ 8	
□ 9	
□ 10	
□ 11	
□ 12	
□ 13	
□ 14	
□ 15	
□ 16	
□ 17	
	付録

# PowerShellのインストール

## III PowerShellのインストールについて

PowerShellは、Windowsのバージョンによっては、標準ではインストールされていません。その場合は、PowerShell(Windows Management Framework)と対応する.NET Frameworkのパッケージをダウンロードしてインストールする必要があります。また、標準でインストールされている場合でも、上位バージョンをインストールしてアップグレードすることができます。

PowerShellをインストールする場合は、最新のサービスパック(SP)を適用し、インストール可能な最新バージョンのPowerShellをインストールすることをお勧めします。なお、環境によっては、古いバージョンのPowerShellをインストールしても、自動的に最新のPowerShellにアップグレードされる場合があります。

### ▶ PowerShellとWindowsのバージョンの対応

PowerShellとWindowsのバージョンの対応は、次の表のとおりです。

OS	標準装備のPowerShellのバージョン	インストール可能なPowerShellのバージョン
Windows Server 2003/2003 R2	なし	1.0(SP1)、2.0(SP2)
Windows Server 2008	1.0	2.0(SP1)、3.0(SP2)
Windows Server 2008 R2	2.0	3.0/4.0(SP1)/5.0(SP1)
Windows Server 2012	3.0	4.0/5.0
Windows Server 2012 R2	4.0	5.0
Windows XP x64	なし	1.0
Windows XP	なし	1.0(SP2)、2.0(SP3)
Windows Vista	なし	1.0、2.0(SP1)
Windows 7	2.0	3.0/4.0(SP1)/5.0(SP1)
Windows 8	3.0	なし
Windows 8.1	4.0	5.0
Windows 10	5.0	なし

※()内は、インストールする際に適応する必要があるサービスパック

Windows Server 2008では、サーバーマネージャーの「機能の追加」でPowerShellを有効化する必要があります。また、Windows Server 2008 R2で、PowerShellのGUI環境(ISEなど)を利用するには、サーバーマネージャーの「機能の追加」で、「.NET Framework 3.5.1」と「Windows PowerShell Integrated Scripting Environment(ISE)」を有効にする必要があります。

各バージョンのPowerShell(Windows Management Framework)の入手先(ダウンロード先)のURLは、次の表のとおりです。

バージョン	URL
1.0(Vista用)	<a href="http://support.microsoft.com/kb/928439">http://support.microsoft.com/kb/928439</a>
2.0	<a href="http://support.microsoft.com/kb/968929">http://support.microsoft.com/kb/968929</a>
3.0	<a href="https://www.microsoft.com/en-us/download/details.aspx?id=34595">https://www.microsoft.com/en-us/download/details.aspx?id=34595</a>
4.0	<a href="https://www.microsoft.com/ja-jp/download/details.aspx?id=40855">https://www.microsoft.com/ja-jp/download/details.aspx?id=40855</a>
5.0	<a href="https://www.microsoft.com/en-us/download/details.aspx?id=46889">https://www.microsoft.com/en-us/download/details.aspx?id=46889</a>

※2015年8月現在、XPおよびServer 2003用のPowerShell 1.0は提供されていません。

### ▶ PowerShellと.NET Frameworkの対応

PowerShellを動作させるには、各バージョンに対応する.NET Frameworkをあらかじめインストール必要があります。PowerShellと.NET Frameworkのバージョンは、次の表のとおりです。

PowerShellのバージョン	.NET Framework
1.0	2.0以上
2.0	2.0以上 (GUI環境の動作には3.5以上)
3.0	4.0以上
4.0	4.5以上
5.0	4.5以上

各バージョンの.NET Frameworkの入手先(ダウンロード先)のURLは、次の表のとおりです。

バージョン	URL
2.0	<a href="http://www.microsoft.com/ja-jp/download/details.aspx?id=6523">http://www.microsoft.com/ja-jp/download/details.aspx?id=6523</a>
3.5	<a href="http://www.microsoft.com/ja-jp/download/details.aspx?id=21">http://www.microsoft.com/ja-jp/download/details.aspx?id=21</a>
4.0	<a href="http://www.microsoft.com/ja-jp/download/details.aspx?id=17718">http://www.microsoft.com/ja-jp/download/details.aspx?id=17718</a>
4.5	<a href="http://www.microsoft.com/ja-jp/download/details.aspx?id=30653">http://www.microsoft.com/ja-jp/download/details.aspx?id=30653</a>

## PowerShellの起動

### III [スタート]メニューからの起動(Windows Vista/7/10/Server 2008)

「スタート」メニューからPowerShellを起動するには、次のように操作します。

- ① [スタート]ボタンをクリックし、[すべてのアプリ]→[Windows PowerShell]→[Windows PowerShell] (Vista/7/Server 2008は[すべてのプログラム]→[アクセサリ]→[Windows PowerShell])を選択します。
  - ② コンソール版の場合は[Windows PowerShell]を、ISE(GUI版)の場合は[Windows PowerShell ISE]をクリックします(「(x86)」と付いているメニューをクリックすると、32bit版が起動)。

### ● Windows 10の例



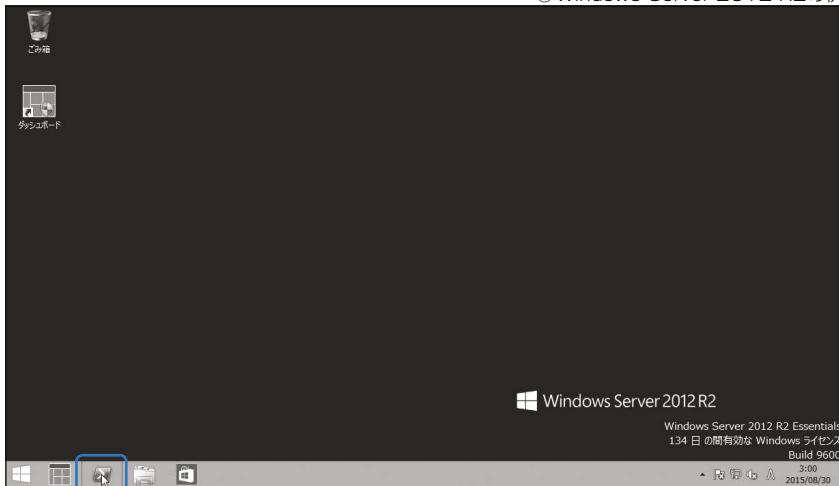
なお、PowerShellを管理者として起動するには、操作②のタイミングで[Windows PowerShell] (ISEは[Windows PowerShell ISE])を右クリックし、[管理者として実行] (Vista/7/Server 2008は、[管理者として実行(A)])を選択します。

### III タスクバーからの起動(Windows 7/Server 2008 R2以降)

タスクバーから、PowerShell(コンソール版)を起動するには、タスクバーの[Windows PowerShell]アイコンをクリックします。ISE(GUI版)を起動するには、タスクバーの[Windows PowerShell]アイコンを右クリックし、[Windows PowerShell ISE]をクリックします。また、管理者として実行するには、タスクバーの[Windows PowerShell]アイコンを右クリックし、[管理者として実行する](ISEの場合は、[ISEを管理者として実行する])をクリックします。

なお、Windows 7/8/8.1では、環境によってはタスクバーに[Windows PowerShell]アイコンがない場合があります。

## ◎Windows Server 2012 R2の例

□1  
□2  
□3  
□4  
□5  
□6  
□7  
□8  
□9  
□10  
□11  
□12  
□13  
□14  
□15  
□16  
□17  
付録CHAPTER  
PowerShellの基礎知識

## III スタート画面からの起動(Windows 8/Server 2012以降)

スタート画面からPowerShell(コンソール版)を起動するには、アプリ一覧を表示(Windows 8/Server 2012では、スタート画面の空欄を右クリック、Windows 8.1ではマウスを動かしたときに画面左下に表示される[↓]をクリック)し、[Windows PowerShell]をクリックします。Server 2012では、スタート画面の[Windows PowerShell]タイルをクリックして、PowerShell(コンソール版)を起動することもできます。

Windows 8でアプリ一覧にISEや32bit版のPowerShellを起動するためのアイコン(タイル)を常に表示するには、スタート画面もしくはアプリ一覧のチャームから[設定]→[タイル]を選択し、[管理ツールを表示する]を[はい]に設定します。

## ◎Windows 8.1の例



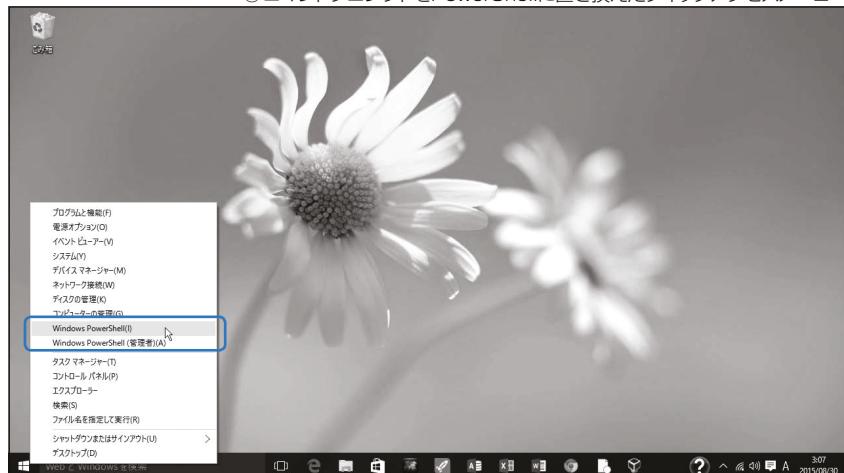
なお、管理者として実行するには、[Windows PowerShell] (ISEは[Windows PowerShell ISE])を右クリックします。Windows 8/Server 2012の場合は、画面下に表示されるアプリバーの[管理者として実行]をクリックします。Windows 8.1の場合は、表示されるショートカットメニューから[管理者として実行(A)]を選択します。

### III クイックアクセスメニューから起動する(Windows 8.1/10)

クイックアクセスメニューからPowerShell(コンソール版)を起動するには、次のように操作して、クイックアクセスメニューのコマンドプロンプトをPowerShellに置き換えます。

- ① タスクバーを右クリックし、[プロパティ(R)]を選択する
- ② 「ナビゲーション」タブを選択し、[右下隅を右クリックするか、Windowsキー + Xキーを押したときに表示されるメニューで、コマンドプロンプトをWindows PowerShellに置き換える(C)]のチェックボックスをONにし、[OK]ボタンをクリックします。

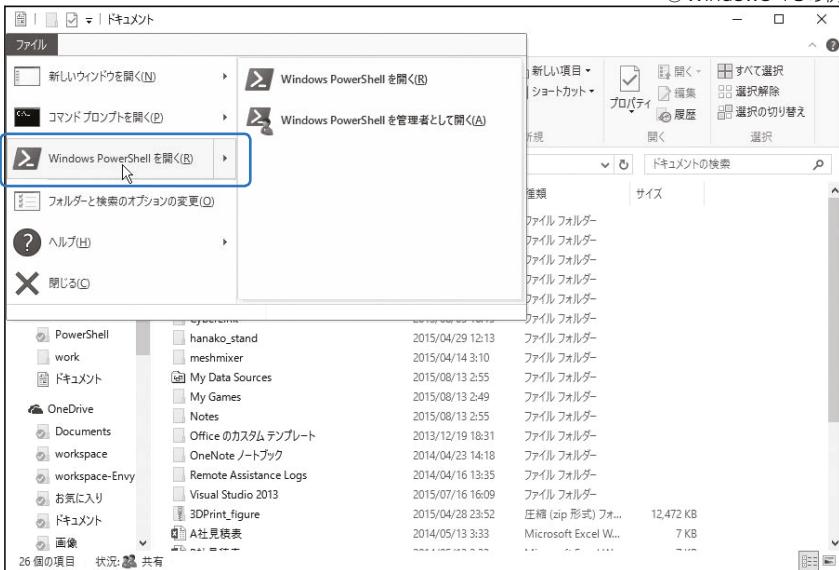
●コマンドプロンプトをPowerShellに置き換えたクイックアクセスメニュー



### III エクスプローラから起動する(Windows 8/Server 2012以降)

エクスプローラからPowerShell(コンソール版)を起動するには、任意のドライブ、フォルダを開いて、[ファイル]→[Windows PowerShellを開く(R)] (または[ファイル]→[Windows PowerShellを開く(R)]→[Windows PowerShellを開く(R)])をクリックします。管理者としてWindows PowerShellを開く場合は、[ファイル]→[Windows PowerShellを開く(R)]→[管理者としてWindows PowerShellを開く(A)]を選択します。

## ● Windows 10の例



## III サーバーマネージャーから起動する(Windows Server 2012以降)

サーバーマネージャーから、「ツール(T)」メニューから[Windows PowerShell] (32bit版は[Windows PowerShell (x86)])を選択します。ISE(GUI版)を起動するには、「ツール(T)」メニューから[Windows PowerShell ISE] (32bit版は[Windows PowerShell ISE (x86)])を選択します。サーバーマネージャーからは管理したいサーバーを選択し、そのサーバーにリモート接続した状態でPowerShellを起動することができます。

## ● Windows Server 2012 R2の例



# コマンドレットについて

## III コマンドレットとは

コマンドレットとは、PowerShellで利用可能なコマンド(命令)のことです。コマンドレットは単独で実行することも、パイプラインで複数を組み合わせて実行することができます(49ページ参照)。また、複数のコマンドレットの組み合わせをスクリプトとして記述することができます。各バージョンのPowerShellに標準で用意されているコマンドレット(PowerShellのコアモジュールに含まれるコマンドレット)の数は、次の表のとおりです。なお、3.0、4.0、5.0は「Core」「Diagnostic」「Host」「Management」「Security」「Utility」「WS-Management」モジュールのコマンドレットの総数になります。

バージョン	コマンドレット数
1.0	129
2.0	236
3.0	259
4.0	262
5.0	275

これ以外にもモジュールにて、多数のコマンドレットが提供されます。利用可能なコマンドレットを調べるには、「Get-Command」コマンドレットを使います(129ページ参照)。

## III コマンドレットの命名規則

PowerShellのコマンドレットは、動作を容易に推測できるように、次のような標準的な名前付けパターンを使用しています。

[動詞]-[名詞]

たとえば、「Get-Location」は、「Location(ロケーション)をGet(取得)する」という意味で、文字通り、ロケーション(カレントディレクトリ)を取得するコマンドレットになります。なお、PowerShellでは、大文字・小文字を区別しないので、「get-location」や「Get-location」と入力、または記述しても構いません。

# コマンドレットのパラメータについて

## III コマンドレットのパラメータ

コマンドレットには、複数のパラメータが指定される場合があります。「Get-Help」コマンドレットを実行すると、コマンドレットのすべてのパラメータが表示されます。

パラメータに関する追加情報は、パラメータの説明およびパラメータ属性テーブルにあります。パラメータの詳細な情報を表示するには、「Get-Help」の「Full」パラメータまたは「Parameter」パラメータを使用します(120ページ参照)。

## III ヘルプの構文

PowerShellのコマンドレットヘルプ、ヘルプトピック、およびその他のドキュメントでは、構文の説明において、コマンドレットで次の表記法が使用されます。

```
<コマンドレット名> -<必須パラメータ名> <必須パラメータ値>
    [-<オプションパラメータ名> <オプションパラメータ値>]
    [-<オプションスイッチパラメータ>]
    [-<オプションパラメータ名>] <必須パラメータ値>
```

たとえば、「New-Alias」コマンドレットの構文は次のとおりです。

```
New-Alias [-Force] [-PassThru] [-Scope <string>]
    [-Option {None | ReadOnly | Constant | Private | AllScope}]
    [-Description <string>] [-Name] <string> [-Value] <string>
    [-confirm] [-whatif] [<CommonParameters>]
```

なお、この構文では、読みやすさのために大文字が使用されていますが、PowerShellでは大文字と小文字は区別されません。

パラメータは順に表示されます。パラメータの順序は、そのパラメータ名がオプションである場合にのみ重要です。コマンドレットの使用時にパラメータ名を指定しない場合、PowerShellでは位置および型に基づいてパラメータに値が割り当てられます。

パラメータ名の前には「-」(ハイフン)が付きます。スイッチパラメータは、値の型を伴わずに表示されます。その他のパラメータは、必須引数のMicrosoft .NET Framework型または使用可能な値の列挙とともに表示されます。

「<>」(山カッコ)はプレースホルダーテキストを示します。このテキストでは、項目の種類(文字列やオブジェクトなど)を記述することができます。このテキストは1つまたは複数の共通パラメータのプレースホルダーである場合もあります。

「[]」(角カッコ)はオプションの項目を示します。パラメータが省略可能であるか、必須パラメータの名前が省略可能であることを示します。スイッチパラメータは、常に省略可能です。

□1	CHAPTER
□2	PowerShellの基礎知識
□3	
□4	
□5	
□6	
□7	
□8	
□9	
10	
11	
12	
13	
14	
15	
16	
17	
付録	

「[]」(波カッコ)は、列挙型の値の前後に表示されます。「New-Alias」コマンドレットの例では、「Option」パラメータには、一覧表示されているいずれの値(「None」「ReadOnly」「Constant」「Private」「AllScope」のいずれか)を指定することもできます。

### III オプションの項目

オプションの項目は、「[]」(角カッコ)で囲まれています。たとえば、「New-Alias」コマンドレットの構文では、「Scope」パラメータはオプションです。これはパラメータ名と型を「[]」が囲む構文で示されます。

```
[-Scope <string>]
```

たとえば、次の2つの「New-Alias」コマンドレットの例は、どちらも正しい使用方法です。

```
New-Alias -Name utd -Value Update-TypeData
New-Alias -Name utd -Value Update-TypeData -Scope global
```

パラメータの値が必須であっても、パラメータ名は省略可能の場合があります。これは、「New-Alias」コマンドレットの次の例のように、パラメータ名を「[]」で囲み、パラメータの型は「[]」で囲まない構文で示します。

```
[-Name] <string> [-Value] <string>
```

たとえば、次の4つのコマンドの結果は同じになります。

```
New-Alias -Name utd -Value Update-TypeData
New-Alias -Name utd Update-TypeData
New-Alias utd -ValueUpdate-TypeData
New-Alias utd Update-TypeData
```

入力時にパラメータ名がステートメントに含まれていない場合、PowerShellは、その引数の位置を使用して値をパラメータに割り当てようとします。たとえば、次の例は、「Value」パラメータが指定されていないので不完全です。

```
New-Alias utd
```

「New-Alias」コマンドレットでは、「Name」および「Value」パラメータの両方に値を指定する必要があります。

構文の例では、「[]」は.NET Framework型に対する名前付けおよびキャストにも使用されています。このような場合、「[]」は要素が省略可能であることを示すわけではありません。

### III 配列引数

「[]」(角カッコ)は配列を表すためにも使用されています。たとえば、「Restart-Service」コマンドレットの構文は、次のようになります。

```
Restart-Service [-Name] <string[]> [-Include <string[]>]  
[-Exclude <string[]>] [-Force] [-PassThru] [-Confirm] [-WhatIf]  
[<CommonParameters>]
```

「Name」パラメータには、再起動するサービスの名前が引数として必要です。この構文で引数は「string[]」と、配列で指定できることを表しているので、次のようにサービスの名前をコンマ区切り一覧で指定することができます。

```
Restart-Service RasAuto, RasMan, RemoteAccess
```

CHAPTER	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	付録
---------	---	---	---	---	---	---	---	---	---	----	----	----	----	----	----	----	----	----

### III 共通パラメータ

共通パラメータ(CommonParameters)は、すべてのコマンドレットで使用できるパラメータです。共通パラメータはどのコマンドレットでも使用できますが、一部のコマンドレットに対しては効力がない場合があります。共通パラメータの詳細は、次のとおりです(())内はエイリアス)。

## ▶ Verbose(vb)

「Verbose(vb)」は、コマンドによって実行される操作の詳細メッセージを表示するかどうかを指定します。このパラメータはコマンドに「Write-Verbose」コマンドレットが含まれている場合に有効になります。「\$true」、または値を指定しない場合は、詳細メッセージを表示します。「\$false」(既定値)の場合は、詳細メッセージを表示しません。

構文は次のとおりです。

-Verbose[:{\$true | \$false}]

## ▶ Debug(db)

「Debug(db)」は、コマンドによって実行される操作のデバッグメッセージ(プログラマレベルの詳細情報)を表示するかどうかを指定します。このパラメータはコマンドに「Write-Debug」コマンドレットが含まれている場合に有効になります。「\$true」または値を指定しない場合は、デバッグメッセージを表示します。「\$false」の場合は、「\$DebugPreference」の値が既定値(SilentlyContinue)でない場合にデバッグメッセージを表示しません。

構文は次のとおりです。

-Debug[·{\$true | \$false}]

### ► WarningAction(wa)

「WarningAction(wa)」は、警告が発生したときの処理方法を指定します。このパラメータはコマンドに「Write-Warning」コマンドレットが含まれている場合に有効になります。

構文は次のとおりです。

-WarningAction{:SilentlyContinue | Continue | Inquire | Stop}

設定可能な値は、次のとおりです。

値	説明
SilentlyContinue	警告メッセージが表示されないようにし、コマンドの実行を継続する
Continue	警告メッセージを表示し、コマンドの実行を継続する(既定値)
Inquire	警告メッセージを表示し、実行を継続する前に確認メッセージを表示する
Stop	警告メッセージを表示し、コマンドの実行を停止する

### ▶ WarningVariable(wv)

「WarningVariable(wv)」は、コマンドに関する警告を変数に格納します。変数の内容に警告を追加するには、変数名の前に「+」(プラス記号)を入力します。

構文は次のとおりです。

```
-WarningVariable [+]<変数名>
```

### ▶ ErrorAction(ea)

「ErrorAction(ea)」は、エラーが発生したときの処理方法を指定します。このパラメータはコマンドに「Write-Error」コマンドレットが含まれている場合に有効になります。

構文は次のとおりです。

```
-ErrorAction[:{Continue | Ignore | Inquire | SilentlyContinue | Stop | Suspend}]
```

設定可能な値は、次のとおりです。

値	説明
Continue	エラーメッセージを表示し、コマンドの実行を継続する(既定値)
Ignore(V3~)	エラーメッセージを表示せずに、コマンドの実行を継続する
Inquire	エラーメッセージを表示し、実行を継続する前に確認メッセージを表示する
SilentlyContinue	エラーメッセージが表示されないようにし、コマンドの実行を継続する
Stop	エラーメッセージを表示し、コマンドの実行を停止する
Suspend(V4~)	エラーメッセージを表示し、コマンドを中断する(ワークフローのみサポート)

### ▶ ErrorVariable(ev)

「ErrorVariable(ev)」は、コマンドに関するエラーメッセージを指定された変数および「\$Error」自動変数に格納します。変数の内容にエラーメッセージを追加するには、変数名の前に「+」(プラス記号)を入力します。

構文は次のとおりです。

```
-ErrorVariable [+]<変数名>
```

### ▶ OutVariable(ov)

「OutVariable(ov)」は、コマンドからの出力オブジェクトを指定された変数に格納し、コマンドラインに表示します。変数に出力を追加するには、変数名の前に「+」(プラス記号)を入力します。

構文は次のとおりです。

```
-OutVariable [+]<変数名>
```

### ▶ OutBuffer(ov)

「OutBuffer(ov)」は、パイプライン経由でオブジェクトが送信されるまでに、バッファーに蓄積するオブジェクト数を決定します。

構文は次のとおりです。

CHAPTER
PowerShellの基礎知識
□ 1
□ 2
□ 3
□ 4
□ 5
□ 6
□ 7
□ 8
□ 9
□ 10
□ 11
□ 12
□ 13
□ 14
□ 15
□ 16
□ 17
付録

```
-OutBuffer <整数値>
```

#### ▶ PipelineVariable(pv)

「PipelineVariable(pv)」は、現在のパイプラインの要素の値を指定された変数に格納します。

構文は次のとおりです。

```
-PipelineVariable <変数名>
```

#### ▶ InformationAction(infa)

「InformationAction(infa)」は、情報メッセージの処理方法を指定します(V5のみ)。このパラメータはコマンドに「Write-Information」コマンドレットが含まれている場合に有効になります。

構文は次のとおりです。

```
-InformationAction [:{SilentlyContinue | Continue | Inquire | Stop | Ignore}]
```

設定可能な値は、次のとおりです。

値	説明
Continue	情報メッセージを表示し、コマンドの実行を継続する(既定値)
Ignore	情報メッセージを表示せずに、コマンドの実行を継続する
Inquire	情報メッセージを表示せずに、実行を継続する前に確認メッセージを表示する
SilentlyContinue	情報メッセージを表示されないようにし、コマンドの実行を継続する
Stop	情報メッセージを表示し、コマンドの実行を停止する

#### ▶ InformationVariable(iv)

「InformationVariable(iv)」は、コマンドに関する情報メッセージを指定された変数に格納します。変数の内容に情報メッセージを追加するには、変数名の前に「+」(プラス記号)を入力します(V5のみ)。

構文は次のとおりです。

```
-InformationVariable [+]<変数名>
```

### III リスク軽減パラメータ

リスク軽減パラメータは、コマンド実行時のリスクを回避または軽減するためのパラメータです。リスク軽減パラメータは、次のとおりです（()内はエイリアス）。

#### ▶ WhatIf(wi)

「WhatIf(wi)」は、コマンドを実行する代わりにコマンドの結果を説明するメッセージを表示します。コマンドを実際に実行することなく、コマンドの動作を確認できます。「\$true」または値を指定しない場合は「WhatIf」の動作を有効にし、「\$false」の場合は、「\$WhatIfPreference」変数の値が「1」であった場合に発生する自動的な「WhatIf」の動作を抑止します。

構文は次のとおりです。

```
-WhatIf[{:{$true | $false}]
```

#### ▶ Confirm(cf)

「Confirm(cf)」は、コマンドを実行する前に確認メッセージを表示します。「\$true」または値を指定しない場合は、「Confirm」の動作を有効にし、「\$false」の場合は、「\$ConfirmPreference」の値がコマンドレットの推定リスク以下であった場合に表示される自動確認メッセージを抑止します。

構文は次のとおりです。

```
-Confirm[{:{$true | $false}]
```

□ 1	PowerShellの基礎知識
□ 2	
□ 3	
□ 4	
□ 5	
□ 6	
□ 7	
□ 8	
□ 9	
1 □	
1 1	
1 2	
1 3	
1 4	
1 5	
1 6	
1 7	
	付録

# エイリアスについて

## III エイリアスとは

エイリアスとは、コマンドレットやコマンドの要素(関数、スクリプト、ファイル、実行可能ファイルなど)に使用する代替名です。PowerShellのすべてのコマンドでは、エイリアスをコマンド名の代わりに使用できます。エイリアスを作成するには、「New-Alias」コマンドレットを使用します(136ページ参照)。

また、PowerShellには、「Set-Location」コマンドレットに関連付けられた「cd」と「chdir」や、「Get-ChildItem」コマンドレットに関連付けられた「ls」と「dir」などの一連の組み込みエイリアスが用意されています。定義されているすべてのエイリアス(組み込みエイリアスを含む)を取得するには、「Get-Alias」を使用します(134ページ参照)。

組み込みエイリアスの例(抜粋)を下表に示します。

コマンドレット	エイリアス
Get-Content	cat, gc
Set-Location	cd, chdir, sl
Copy-Item	copy, cp, cpi
Get-ChildItem	dir, gci, ls
Get-Location	gl, pwd
Move-Item	mi, move, mv
Remove-Item	rd, ri, rm, rmdir
Rename-Item	ren, rni

1つのコマンドレットには、複数の組み込みエイリアスが関連付けられている場合があります。組み込みエイリアスには、そのコマンドレットの同等の機能を持つコマンドプロンプトのコマンド(「cd」「dir」など)やUNIXのシェルコマンド(「cat」「ls」など)、またはコマンドレットの動詞、名詞部分から一部の文字列を抜粋(多くの場合、それぞれの頭文字)した名前(「gc」「sl」「gci」など)が使用されています。

# リダイレクトについて

## リダイレクトとは

リダイレクトとは、コマンド(コマンドレット)の出力先を別の場所に変更することです。既定では、PowerShellのコマンド出力は、PowerShellコンソールに送信されます。ただし、出力をテキストファイルに送信することも、エラー出力を通常の出力ストリームにリダイレクトすることもできます。

## リダイレクトの方法

PowerShellでは、次の方法で出力をリダイレクトできます。

### 「Out-File」コマンドレットを使用する

「Out-File」コマンドレットを使用し、コマンド出力をテキストファイルに送信します。通常、「Out-File」コマンドレットは、そのパラメータ(「Encoding」「Force」「Width」「NoClobber」など)を使用する必要がある場合に使用します(272ページ参照)。

### 「Tee-Object」コマンドレットを使用する

「Tee-Object」コマンドレットは、コマンド出力をテキストファイルに送信し、その後でパイプラインに送信します(334ページ参照)。

### リダイレクト演算子を使う

PowerShellのリダイレクト演算子は、次のとおりです。

演算子	説明	例
>	指定したファイルに出力を送信する	Get-Process > process.txt
>>	指定したファイルの内容に出力を追加する	dir *.ps1 >> scripts.txt
n>	指定したファイルにエラーを送信する	Get-Process none 2> errors.txt
n>>	指定したファイルの内容にエラーを追加する	Get-Process none 2>> save-errors.txt
n>&1	正常な出力ストリームにエラーを送信する	Get-Process none, powershell 2>&1

上記の表中の「n」には、次のストリーム番号を指定できます。

ストリーム番号	ストリーム
2	エラー
3(V3~)	警告
4(V3~)	詳細
5(V3~)	デバッグ
*(V3~)	すべての出力

リダイレクト演算子の構文は次のとおりです。

<入力コマンド> <演算子> [<バス>]<ファイル名>

CHAPTER 01
□ 02
□ 03
□ 04
□ 05
□ 06
□ 07
□ 08
□ 09
□ 10
□ 11
□ 12
□ 13
□ 14
□ 15
□ 16
□ 17
付録

指定したファイルがすでに存在する場合、データを追加しないリダイレクト演算子（「>」および「2>」）を使用すると、ファイルの現在の内容が警告なしに上書きされます。ただし、ファイルが読み取り専用ファイル、非表示のファイル、システムファイルのいずれかの場合、リダイレクトは失敗します。追加リダイレクト演算子（「>>」および「2>>」）を使用した場合、読み取り専用ファイルには書き込みは行われませんが、システムファイルと非表示のファイルには内容が追加されます。

読み取り専用ファイル、非表示のファイル、またはシステムファイルに対し、内容のリダイレクトを強制的に行うには、「Out-File」コマンドレットとその「Force」パラメータを使用します。リダイレクト演算子は、ファイルへの書き込み時にUnicodeエンコードを使用します。ファイルのエンコードが異なる場合、出力が適切にフォーマットされないことがあります。内容を非Unicodeファイルにリダイレクトするには、「Out-File」コマンドレットとその「Encoding」パラメータを使用します（272ページ参照）。

# パイプラインについて

## III パイプラインとは

パイプラインとは、「|」(パイプライン演算子)を使用して連結された一連のコマンドです。各パイプライン演算子は、直前のコマンドの結果を次のコマンドに渡します。

パイプラインを使用すると、一方のコマンドにより出力されたオブジェクトを他方のコマンドの入力として使用できます。さらに、そのコマンドの出力をさらに別のコマンドに渡すこともできます。その結果、一連の単純なコマンドから、非常に強力なコマンドチェーンを作成できます。

パイプラインの例は、次のようにになります。

```
コマンド1 | コマンド2 | コマンド3
```

パイプラインでは、左から右に、表示された順序でコマンドが処理されます。上の例では、コマンド1により出力されるオブジェクトがコマンド2に渡されます。コマンド2は、これらのオブジェクトを処理し、コマンド3に渡します。コマンド3は、これらのオブジェクトを処理し、パイプラインでの次の処理に渡します。このパイプラインにはこれ以上コマンドが存在しないので、結果がコンソールに表示されます。

なお、パイプラインによる処理(一方のコマンドの出力に含まれるオブジェクトが他方のコマンドに渡される処理)を、「パイプ処理」といいます。

## III PowerShellでのパイプラインの使用

PowerShellのコマンドレットは、パイプラインでの使用を想定して設計されています。たとえば、一般的な使用例では、「Get」コマンドレットで取得した結果をパイプ処理し、同じ名詞の処理コマンドレット(「Set」「Start」「Stop」「Rename」コマンドレットなど)に渡すことができます。

たとえば、「Get-Service」コマンドレットからすべてのサービスをパイプ処理して、「Start-Service」、または「Stop-Service」コマンドレットに渡すことができます。次のパイプラインを実行すると、コンピュータ上でWMIサービスが起動されます。

```
Get-Service wmi | Start-Service
```

PowerShellプロバイダーのオブジェクトの取得または設定を行うコマンドレット(「Item」および「ItemProperty」コマンドレットなど)も、パイプラインで使用できるように設計されています。たとえば、PowerShellレジストリプロバイダーの「Get-Item」または「Get-ChildItem」コマンドレットの結果をパイプ処理して、「New-ItemProperty」コマンドレットに渡すことができます。

たとえば、「HKLM:¥Software¥MyCompany」レジストリキーに「8124」という値を持つ新しいレジストリエントリ「NoOfEmployees」を追加するには、次のように入力します。

```
Get-Item -Path HKLM:¥Software¥MyCompany | New-ItemProperty -Name NoOfEmployees -Value 8124
```

CHAPTER 01
□ 02
□ 03
□ 04
□ 05
□ 06
□ 07
□ 08
□ 09
□ 10
□ 11
□ 12
□ 13
□ 14
□ 15
□ 16
□ 17
付録

「Get-Member」「Where-Object」「Sort-Object」「Group-Object」および「Measure-Object」などの多くのユーティリティコマンドレットは、ほぼパイプラインにのみ使用されます。これらのコマンドレットには、あらゆるオブジェクトをパイプ処理で渡すことができます。たとえば、コンピュータ上のすべてのプロセスをパイプ処理して、「Sort-Object」コマンドレットに渡し、それらをハンドル数に基づいて並べ替えることができます。

```
Get-Process | Sort-Object -Property handles
```

また、「Format-List」および「Format-Table」などの書式設定用コマンドレット、「Export-Clixml」および「Export-Csv」などの「Export」コマンドレット、「Out-Printer」などの「Out」コマンドレットにも、任意のオブジェクトをパイプ処理で渡すことができます。たとえば、「Winlogon」プロセスを「Format-List」コマンドレットにパイプ処理して渡し、プロセスのすべてのプロパティを一覧表示できます。

```
Get-Process winlogon | Format-List -Property *
```

### III PowerShellでのパイプラインの動作

オブジェクトがパイプ処理される（一方のコマンドの出力に含まれるオブジェクトが他方のコマンドに渡される）と、PowerShellは、パイプ処理されたオブジェクトを受け取り側のコマンドレットのいずれかのパラメータに関連付けます。このため、PowerShellには入力オブジェクトをコマンドレットパラメータに関連付ける（パラメータをバインドする）コンポーネントが存在し、このコンポーネントは、次の条件を満たすパラメータを検索します。

- パイプラインからの入力を許可するパラメータ（すべてのパラメータが許可するわけではないため）
- 渡されるオブジェクトの型またはオブジェクトの変換が可能な目的の型を受け入れるパラメータ
- コマンドでまだ使用されていないパラメータ

たとえば、「Start-Service」コマンドレットには多くのパラメータが存在しますが、パイプラインの入力を許可するのは「Name」および「InputObject」の2つのパラメータのみです。「Name」パラメータは文字列を受け取り、「InputObject」パラメータはサービスオブジェクトを受け取ります。このため、文字列およびサービスオブジェクト（またはサービスオブジェクトに変換可能なプロパティを持つオブジェクト）をパイプ処理して、「Start-Service」に渡すことができます。

PowerShellのパラメータバインドコンポーネントが、パイプ処理されたオブジェクトを受け取り側のコマンドレットパラメータに関連付けることができない場合、コマンドは失敗し、見つからなかったパラメータ値の入力を求めるメッセージがPowerShellによって表示されます。

なお、「Get-Help」コマンドレットを「Full」または「Parameter」パラメータと一緒に使用すると、パイプライン入力を許可するコマンドレットのパラメータを特定することができます（120ページ参照）。

# ワイルドカードについて

## III ワイルドカードとは

ワイルドカード(ワイルドカード文字)とは、任意の文字を表すための記号です。ワイルドカードを使うと、特定のパターンに一致するファイルだけを検索することができます。PowerShellでサポートされるワイルドカードは、次のとおりです。

ワイルドカード	説明	例	一致する	一致しない
*	0個以上の文字と一致する	a*	A, ag, Apple	banana
?	指定された位置にある1文字と正確に一致する	?n	an, in, on	ran
[ ]	指定された文字と一致する	[bc]ook	book, cook	hook
[ - ]	ある範囲の文字と一致する	[a-l]ook	book, cook, look	took

大部分のコマンドレットは、一部のパラメータでワイルドカード文字を受け入れます。各コマンドレットのヘルプトピックには、ワイルドカード文字を使用できるパラメータについての説明があります(該当するパラメータがある場合)。ワイルドカード文字を使用できるパラメータでは、大文字と小文字は区別されません。たとえば、「?n\_」は「An」「an」「In」「in」「On」「on」を返します。

また、1つのパラメータ内で異なるワイルドカード文字を組み合わせることができます。たとえば、「C:\Techdocs」ディレクトリにあるファイルの中で、「a」から「l」までの文字で始まるすべての「.txt」ファイルを表示するには、次のようにコマンドを入力します。

```
Get-ChildItem c:\Techdocs\*[a-l]*.txt
```

このコマンドは、文字範囲を表すワイルドカード「[a-l]」を使用して、ファイル名が「a」から「l」の範囲内の文字で始まる必要があることを示しています。さらに、ワイルドカード「\*」を使用して、最初の文字とファイル拡張子の間には任意の文字列があってよいことを示すプレースホルダーを提供しています。

CHAPTER 01
□ 02
□ 03
□ 04
□ 05
□ 06
□ 07
□ 08
□ 09
□ 10
□ 11
□ 12
□ 13
□ 14
□ 15
□ 16
□ 17
付録

# スクリプトについて

## III スクリプトとは

スクリプトとは、1つまたは複数のPowerShellコマンド(コマンドレット)を含むテキストファイルです。PowerShellのスクリプトのファイル名拡張子は「.ps1」です。スクリプトを作成すると、スクリプトのパスとファイル名を入力するだけでコマンドを実行でき、コマンドを再入力する手間を省くことができます。また、簡単に他のユーザーとコマンドを共有できます。

スクリプトは、#Requires特殊コメント、パラメータの使用、Dataセクションのサポート、セキュリティを確保するためのデジタル署名など、さまざまな機能を備えています。スクリプトやスクリプト内の任意の関数に対するヘルプトピックを記述することもできます。

## III スクリプトの作成方法

スクリプトを作成するには、テキストエディタ(メモ帳など)またはスクリプトエディタ(ISEなど)を起動します。コマンドを入力し、有効なファイル名と「.ps1」ファイル名拡張子を付けたファイルに保存します。スクリプトには、単独のコマンドに加えて、パイプライン、関数、制御構造(「If」ステートメントや「For」ループなど)を使用するコマンドを含めることができます。

## III スクリプトの実行方法

スクリプトを実行する前に、PowerShellの既定の実行ポリシーを変更する必要があります。既定の実行ポリシーである「Restricted」では、ローカルコンピュータ上に作成したスクリプトを含むすべてのスクリプトの実行が禁止されます(442ページ参照)。

PowerShellのセキュリティ機能により、Windowsエクスプローラでスクリプトアイコンをダブルクリックしても、スクリプトは実行されません。スクリプトを実行するには、スクリプトファイルの完全な名前を含む完全なパスを入力します。たとえば、「C:\\$Scripts」ディレクトリにある「ServicesLog」スクリプトを実行するには、次のように入力します。

```
PS C:\$work> c:\$scripts\$ServicesLog.ps1
```

現在のディレクトリ(カレントディレクト)にあるスクリプトを実行するには、現在のディレクトリへのパスを入力するか、または現在のディレクトリを表すドットに続けて円マーク(¥)を使用します。たとえば、カレントディレクトにある「ServicesLog.ps1」スクリプトを実行するには、次のように入力します。

```
PS C:\$work> .\$ServicesLog.ps1
```

# プロバイダーについて

## III プロバイダーとは

プロバイダー（PowerShellプロバイダー、PSプロバイダー）とは、特殊なデータストアのデータ（PowerShellのエイリアス、関数、変数など）をPowerShellで利用できるようにし、それらのデータの表示と管理を可能にするためのプログラムです。

プロバイダーが公開するデータはドライブに表示されます。このデータには、ハードディスクドライブ上のデータにアクセスする場合と同じように、パスを使用してアクセスします。プロバイダーがサポートする組み込みコマンドレットを使用して、プロバイダーのドライブにあるデータを管理できます。また、このデータ用に特別に設計されたカスタムコマンドレットも使用できます。

## III 組み込みプロバイダー

PowerShellには、さまざまなデータストアへのアクセスに使用できる、組み込みプロバイダーのセットが用意されています。

プロバイダー	ドライブ	データストア
Alias	Alias:	PowerShellエイリアス
Certificate	Cert:	デジタル署名用のX.509証明書
Environment	Env:	Windows環境変数
FileSystem	C:、D:など(システムによって異なる)	ファイルシステムのドライブ、ディレクトリ、およびファイル
Function	Function:	PowerShellの関数
Registry	HKLM:、HKCU:	Windowsレジストリ
Variable	Variable:	PowerShellの変数
WSMan	WSMan:	WS-Management構成情報

また、独自のPowerShellプロバイダーを作成したり、他のユーザーが開発したプロバイダーをインストールしたりすることもできます。セッションで使用可能なプロバイダーの一覧を表示するには、「Get-PSProvider」コマンドレットを使います(240ページ参照)。

## III 「Alias」プロバイダー

「Alias」プロバイダー（エイリアスプロバイダー）は、PowerShellのエイリアスおよびエイリアスで表される値へのアクセスを提供します。「Alias」プロバイダーを使用すると、PowerShellでエイリアスを取得、追加、変更、クリア、および削除できます。「Alias」プロバイダーは、「Alias:」ドライブにデータストアを公開します。

### ▶ コマンドレット

「Alias」プロバイダーは、「Invoke-Item」コマンドレットを除き、名前に「Item」という名詞が含まれるすべてのコマンドレットをサポートします。また、「Get-Content」および「Set-Content」コマンドレットもサポートします。ただし、「ItemProperty」という名詞を含むコマンドレットおよび、あらゆるコマンドレットの「Filter」パラメータはサポートされません。なお、PowerShellには、エ

□1	CHAPTER
□2	PowerShellの基礎知識
□3	
□4	
□5	
□6	
□7	
□8	
□9	
10	
11	
12	
13	
14	
15	
16	
17	
	付録

イリヤスの表示や変更を行うために設計された、次のコマンドレットが用意されています。

コマンドレット	解説
Get-Alias	134ページ
New-Alias	136ページ
Set-Alias	138ページ
Export-Alias	139ページ
Import-Alias	141ページ

### ▶ 動的パラメータ

動的パラメータとは、「PowerShell」プロバイダーによって追加されるコマンドレットパラメータで、プロバイダーに対応したドライブでコマンドレットが使用されている場合のみ利用できます。「Alias:」ドライブでサポートされる動的パラメータは、次のとおりです。

#### ● Options <スコープ項目オプション>

エイリアスの「Options」プロパティの値を決定します。

値	説明
None	オプションなし(既定値)
Constant	エイリアスは削除できず、プロパティは変更できない(エイリアスを作成するときのみ指定可能)
Private	エイリアスは、現在のスコープでのみ表示される(子スコープでは表示されない)
ReadOnly	「Force」パラメータを使用しない限り、エイリアスのプロパティは変更できない(「Remove-Item」を使用すると、エイリアスを削除はできる)
AllScope	エイリアスは、新たに作成されるすべてのスコープにコピーされる
Unspecified	オプションはユーザーによって指定されていない(V3~)

サポートされているコマンドレットは、「New-Item」「Set-Item」です。

## III 「Certificate」プロバイダー

「Certificate」プロバイダー(証明書プロバイダー)は、PowerShell内からX509証明書ストアおよび証明書へのアクセスを提供します。「Certificate」プロバイダーを使用して、証明書の名前空間内を移動し、証明書ストアおよび証明書を表示することができます。また、証明書と証明書ストアのコピー、移動、および削除を行うこともできます。「Certificate」プロバイダーは、証明書の名前空間をPowerShellの「Cert:」ドライブとして公開します。「Cert:」ドライブには次の3つのレベルが存在します。

#### ● 保存場所(Microsoft.PowerShell.Commands.X509StoreLocation)

現在のユーザーおよびすべてのユーザーの証明書をグループ化するための高レベルのコンテナーです。各システムには、「CurrentUser」および「LocalMachine」の保存場所があります。

#### ● 証明書ストア(System.Security.Cryptography.X509Certificates.X509Store)

証明書の保存および管理に使用される物理ストアです。

- X509証明書(System.Security.Cryptography.X509Certificates.X509Certificate2)

それぞれがコンピュータのX509証明書を表しています。証明書は押印によって識別されます。

#### ▶ コマンドレット

「Certificate」プロバイダーは、「Set-Location」「Get-Location」「Get-Item」「Get-ChildItem」「Invoke-Item」コマンドレットをサポートしています。さらに、PowerShellセキュリティスナップイン(Microsoft.PowerShell.Security)には、証明書プロバイダーに加えて、Authenticode署名の取得と設定、および証明書の取得を行うスナップインも含まれています。セキュリティスナップインに含まれているコマンドレットの一覧を参照するには、次のように入力します。

```
Get-Command -Module *security
```

#### ▶ 動的パラメータ

「Cert:」ドライブでサポートされる動的パラメータは、次のとおりです。

#### ● CodeSigningCert

コード署名機関を持つ証明書のみを取得します。

サポートされているコマンドレットは、「Get-Item」「Get-ChildItem」です。

### III 「Environment」プロバイダー

「Environment」プロバイダー(環境プロバイダー)は、Windows環境変数へのアクセスを提供します。「Environment」プロバイダーを使用すると、PowerShellでWindows環境変数を取得、追加、変更、クリア、および削除できます。「Environment」プロバイダーは、「Env:」ドライブにデータストアを公開します。

#### ▶ コマンドレット

「Environment」プロバイダーは、「Invoke-Item」を除き、名前に「Item」という名詞が含まれるすべてのコマンドレットをサポートします。また、「Get-Content」および「Set-Content」コマンドレットもサポートします。ただし、「ItemProperty」という名詞を含むコマンドレット、およびあらゆるコマンドレットの「Filter」パラメータはサポートされません。

### III 「FileSystem」プロバイダー

「FileSystem」プロバイダーは、ファイルおよびディレクトリ(フォルダ)へのアクセスを提供します。「FileSystem」プロバイダーを使用すると、PowerShellでファイルやディレクトリを取得、追加、変更、クリア、および削除できます。

#### ▶ コマンドレット

「FileSystem」プロバイダーは、名前に「Item」「Content」、および「ItemProperty」という名詞が含まれるすべてのコマンドレットをサポートします。また、あらゆるコマンドレットの「Filter」パラメータをサポートします。

□ 1	CHAPTER
□ 2	PowerShellの基礎知識
□ 3	
□ 4	
□ 5	
□ 6	
□ 7	
□ 8	
□ 9	
10	
11	
12	
13	
14	
15	
16	
17	
付録	

## ▶ 動的パラメータ

「FileSystem」プロバイダーのドライブでサポートされる動的パラメータは、次のとおりです。

## ● Encoding &lt;エンコード&gt;

ファイルを読み書きするときの文字エンコーディングを指定します。「Encoding」パラメータの有効値は、705ページ参照してください。

サポートされているコマンドレットは、「Add-Content」「Get-Content」「Set-Content」です。

## ● Delimiter &lt;区切り文字&gt;

ファイルを読み取るときに使用する区切り文字を指定します

サポートされているコマンドレットは、「Get-Content」です。

## ● Wait

ファイルの内容が追加されるときまで待機します。

サポートされているコマンドレットは、「Get-Content」です。

## III 「Function」プロバイダー

「Function」プロバイダー(関数プロバイダー)は、PowerShellで定義されている関数へのアクセスを提供します。「Function」プロバイダーを使用すると、PowerShellで関数やフィルターを取得、追加、変更、クリア、および削除できます。「Function」プロバイダーは、「Function:」ドライブにデータストアを公開します。

## ▶ コマンドレット

「Function」プロバイダーは、「Invoke-Item」を除き、名前に「Item」という名詞が含まれるすべてのコマンドレットをサポートします。また、「Get-Content」および「Set-Content」コマンドレットもサポートします。ただし、「ItemProperty」という名詞を含むコマンドレットおよび、あらゆるコマンドレットの「Filter」パラメータはサポートされません。

## ▶ 動的パラメータ

「Function:」ドライブでサポートされる動的パラメータは、「Alias:」ドライブと同じです(54ページ参照)。

## III 「Registry」プロバイダー

「Registry」プロバイダー(レジストリプロバイダー)は、PowerShellからシステムレジストリのキーと値へのアクセスを提供します。「Registry」プロバイダーを使用すると、PowerShellでレジストリのキーと値を取得、追加、変更、クリア、および削除できます。「Registry」プロバイダーは、「HKLM:」および「HKCU:」ドライブにデータストアを公開します。

## ▶ コマンドレット

「Registry」プロバイダーは、名前に「Item」という名詞が含まれるすべてのコマンドレット(「Item」コマンドレット)をサポートします。レジストリのキーとサブキーを操作する場合は、

「Item」コマンドレットを使用します。また、名前に「ItemProperty」という名詞が含まれるコマンドレット（「ItemProperty」コマンドレット）もサポートします。レジストリの値やデータを操作する場合は、「ItemProperty」コマンドレットを使用します。レジストリプロバイダーでは、名前に「Content」という名詞が含まれるコマンドレットは使用できません。

#### ▶ 動的パラメータ

「Registry」プロバイダーのドライブでサポートされる動的パラメータは次のとおりです。

##### ● Type <レジストリ値の種類>

レジストリに値を保存するときに使用するデータ型を指定するか、レジストリの値のデータ型を識別します。有効値は「.PropertyType」パラメータと同じです（712ページ参照）。

サポートされているコマンドレットは、「Set-Item」「Set-ItemProperty」です。

### III 「Variabl」プロバイダー

「Variabl」プロバイダー（変数プロバイダー）は、PowerShellの変数とその値へのアクセスを提供します。「Variabl」プロバイダーを使用すると、現在のコンソールにあるPowerShellの変数を取得、追加、変更、クリア、および削除できます。「Variabl」プロバイダーは、「Variable:」ドライブにデータストアを公開します。

#### ▶ コマンドレット

「Variabl」プロバイダーは、「Invoke-Item」を除き、名前に「Item」という名詞が含まれるすべてのコマンドレットをサポートします。また、「Get-Content」および「Set-Content」コマンドレットもサポートします。ただし、「ItemProperty」という名詞を含むコマンドレットおよび、あらゆるコマンドレットの「Filter」パラメータはサポートされません。なお、PowerShellには、特に変数の表示と変更を行うために設計された、次のコマンドレットが用意されています。

コマンドレット	解説
Get-Variable	152ページ
New-Variable	154ページ
Set-Variable	156ページ
Clear-Variable	157ページ
Remove-Variable	158ページ

### III 「WSMan」プロバイダー

「WSMan」プロバイダーは、Web Services for Management (WS-Management) の構成情報へのアクセスを提供します。「WSMan」プロバイダーを使用すると、ローカルまたはリモートコンピュータ上のWS-Management構成データを追加、変更、クリア、および削除できます。「WSMan」プロバイダーは、「WSMan:」ドライブにデータストアを公開します。

「WSMan」プロバイダーは、PowerShellのドライブと、WS-Managementの構成設定の論理グループに相当するディレクトリ構造を表示します。これらのグループはコンテナーと呼ばれています。

□1	CHAPTER
□2	PowerShellの基礎知識
□3	
□4	
□5	
□6	
□7	
□8	
□9	
10	
11	
12	
13	
14	
15	
16	
17	
付録	

### ● Client

WS-Managementクライアントのさまざまな設定を構成できます。構成情報はレジストリに保存されます。

### ● Service

WS-Managementサービスのさまざまな設定を構成できます。構成情報はレジストリに保存されます。サービス構成は「サーバー構成」と呼ばれることがあります。

### ● Shell

WS-Managementシェルのさまざまな設定を構成できます。たとえば、リモートシェルアクセスを有効にしたり(AllowRemoteShellAccess)、同時にアクセス可能な最大ユーザー数を設定したり(MaxConcurrentUsers)できます。

### ● Listener

リスナーの作成と構成ができます。リスナーは、WS-Managementプロトコルを実装してメッセージを送受信する管理サービスです。

### ● Plugin

プラグインは、さまざまな機能を提供するためにWS-Managementサービスが読み込んで使用します。既定では、PowerShellはイベント転送プラグイン、PowerShellプラグイン、およびWindows Management Instrumentation(WMI)プロバイダープラグインという3つのプラグインを提供しています。これらのプラグインは、イベント転送、構成、およびWMIアクセスをサポートします。

### ● ClientCertificate

クライアント証明書を作成できます。クライアント証明書は、WS-Managementクライアントが証明書認証を使用するように構成されているときに使用されます。

## ▶ コマンドレット

PowerShellでは、次の「WSMan(WS-Management)」コマンドレットを使用できます。

コマンドレット	解説
Connect-WSMan	570ページ
Disconnect-WSMan	572ページ
Get-WSManInstance	573ページ
New-WSManInstance	575ページ
Remove-WSManInstance	578ページ
Set-WSManInstance	580ページ
Invoke-WSManAction	582ページ
New-WSManSessionOption	584ページ
Enable-WSManCredSSP	586ページ

□ 1	CHAPTER
□ 2	PowerShellの基礎知識
□ 3	
□ 4	
□ 5	
□ 6	
□ 7	
□ 8	
□ 9	
□ 10	
□ 11	
□ 12	
□ 13	
□ 14	
□ 15	
□ 16	
□ 17	
付録	

コマンドレット	解説
Get-WSManCredSSP	588ページ
Disable-WSManCredSSP	590ページ
Set-WSManQuickConfig	591ページ
Test-WSMan	592ページ

また、リスナー、プラグイン(InitializationParameters、Resources、Security) および「ClientCertificate」プロバイダーのパスは、特定の「New-Item」のサポートを提供します。カスタムヘルプを参照するには、適切なパスで「Get-Help New-Item」と入力します。

#### ▶ 動的パラメータ

「WSMan:」ドライブでサポートされる動的パラメータは次のとおりです。

- **Address <アドレス>**

このリスナーが作成されたアドレスを指定します。「Address」パラメータの値はリスナーの作成時に設定されます。

値	説明
*	「*」(ワイルドカード文字)を使用すると、ネットワークインターフェイスカード(NIC)上のすべてのIPアドレスが結合される
IP:<IPアドレス>	「IP:」の後にIPv4の「.」(ドット)で区切られた10進数形式またはIPv6の「:」(コロン)で区切られた16進数形式で有効なIPアドレスを入力する
MAC:	「MAC:」の後に、NICのMACアドレスを入力する

サポートされているコマンドレットは、「Get-Item」です。

- **AllowRemoteShellAccess <ブール値>**

リモートシェルへのアクセスを有効にします「False」に設定した場合は、新しいリモートシェル接続がサーバーによって拒否されます。既定値は「True」です。

サポートされているコマンドレットは、「Get-Item」「Set-Item」です。

- **AllowUnEncrypted <ブール値>**

クライアントコンピュータが暗号化されていないトラフィックを要求することを許可します。既定では、クライアントコンピュータは暗号化されたネットワークトラフィックを要求します。

サポートされているコマンドレットは、「Get-Item」「Set-Item」です。

- **Arguments <引数>**

カスタムシェルに渡す引数文字列とコマンドライン引数を指定します。このパラメータは省略可能です。

サポートされているコマンドレットは、「Get-Item」「Set-Item」です。

- **Basic <ブール値>**

クライアントコンピュータが基本(ベーシック)認証を使用することを許可します。

サポートされているコマンドレットは、「Get-Item」「Set-Item」です。

### ● Capability <操作>

このURIでサポートされる操作を指定します。URIがサポートしている各操作につき1つエントリを作成する必要があります。

値	説明
Create	Create操作がサポートされる。Create操作でこの概念がサポートされている場合は、「SupportFragment」属性が使用される。「SupportFiltering」属性は無効なので、「False」に設定する必要がある。Shell操作もサポートされている場合、この操作はURIに対して無効
Delete	Delete操作がサポートされる。Delete操作でこの概念がサポートされている場合は、「SupportFragment」属性が使用される。「SupportFiltering」属性は無効なので、「False」に設定する必要がある。Shell操作もサポートされている場合、この操作はURIに対して無効
Enumerate	Enumerate操作がサポートされる。「SupportFragment」属性は「Enumerate」操作ではサポートされていないため、「False」に設定する必要がある。「SupportFiltering」属性は有効。プラグインでフィルター処理がサポートされている場合は、この属性を「True」に設定する必要がある。Shell操作もサポートされている場合、この操作はURIに対して無効
Get	Get操作がサポートされる。Get操作でこの概念がサポートされている場合は、「SupportFragment」属性が使用される。「SupportFiltering」属性は無効なので、「False」に設定する必要がある。Shell操作もサポートされている場合、この操作はURIに対して無効
Invoke	Invoke操作がサポートされる。「SupportFragment」属性は「Invoke」操作ではサポートされていないため、「False」に設定する必要がある。「SupportFiltering」属性は無効なので、「False」に設定する必要がある。Shell操作もサポートされている場合、この操作はURIに対して無効
Put	Put操作がサポートされる。Put操作でこの概念がサポートされている場合は、「SupportFragment」属性が使用される。「SupportFiltering」属性は無効なので、「False」に設定する必要がある。Shell操作もサポートされている場合、この操作はURIに対して無効
Subscribe	Subscribe操作がサポートされる。「SupportFragment」属性は「Subscribe」操作ではサポートされていないため、「False」に設定する必要がある。「SupportFiltering」属性は無効なので、「False」に設定する必要がある。Shell操作もサポートされている場合、この操作はURIに対して無効
Shell	Shell操作がサポートされる。「SupportFragment」属性はShell操作ではサポートされていないため、「False」に設定する必要がある。「SupportFiltering」属性は無効なので、「False」に設定する必要がある。他の操作もサポートされている場合、この操作はURIに対して無効。URIに対してShell操作が構成されている場合、「Get」「Put」「Create」「Delete」「Invoke」および「Enumerate」操作は、シェルを管理するため、WS-Management(WinRM)サービス内部で処理される。その結果、プラグインで操作を処理できない

サポートされているコマンドレットは、「Get-Item」「New-Item」「Remove-Item」「Set-Item」です。

### ● CbtHardeningLevel <ポリシー>

認証要求内のチャネルバインドトークンの必要条件ポリシーを設定します。このパラメータの値はHTTPS接続に対してのみ有効です。

値	説明
Strict	チャネルバインドトークンがない要求は、すべて拒否される。この設定では、チャネルバインドトークンの使用によってすべての接続がセキュリティで保護される
Relaxed	要求にチャネルバインドトークンがある場合に、接続がセキュリティで保護される。チャネルバインドトークンがない場合でも、接続は受け入れられる。ただし、チャネルバインドトークンの使用によって防げる攻撃に対して脆弱になる
None	指定したチャネルバインドトークンはすべて無視される

サポートされているコマンドレットは、「Get-Item」「Set-Item」です。

#### ● CertificateThumbprint <証明書捺印>

サーバー証明書の捺印を指定します。この値は、証明書の「Thumbprint」フィールドに含まれている2桁の16進数値の文字列を表します。この処理を実行する権限のあるユーザー/アカウントのデジタル公開キー証明書(X509)を指定します。証明書は、クライアント証明書ベースの認証で使用されます。これらはローカルユーザー/アカウントにのみマッピングでき、ドメインアカウントでは機能しません。証明書の捺印を取得するには、「Cert:」ドライブで「Get-Item」または「Get-ChildItem」コマンドレットを使用します。

サポートされているコマンドレットは、「Clear-Item」「Get-Item」「Set-Item」です。

#### ● Certificate <ブール値>

クライアントを証明書認証に使用できます。WS-Managementクライアントはコンピュータストアで証明書を見つけようとします。コンピュータストアで証明書を見つけることができない場合、クライアントは現在のユーザーストアで見つけようとします。一致する証明書が見つからない場合は、エラーメッセージが表示されます。

サポートされているコマンドレットは、「Get-Item」「Set-Item」です。

#### ● CredSSP <ブール値>

クライアントがCredSSP認証を使用することを許可します。CredSSP認証では、ユーザーが資格情報を委任できます。このオプションは、1台のリモートコンピュータで実行していても、データを他のリモートコンピュータから収集したり、追加のコマンドを他のリモートコンピュータで実行したりするコマンド用に設計されています。

サポートされているコマンドレットは、「Get-Item」「Set-Item」です。

#### ● HTTP <ポート番号>

HTTPが使用されるときにクライアントが使用するポートを指定します。既定では、HTTPはポート80を使用します。1~65535の任意の値を指定できます

サポートされているコマンドレットは、「Get-Item」「Set-Item」です。

#### ● HTTPS <ポート番号>

HTTPSが使用されるときにクライアントが使用するポートを指定します。既定では、HTTPS

□1	CHAPTER
□2	PowerShellの基礎知識
□3	
□4	
□5	
□6	
□7	
□8	
□9	
10	
11	
12	
13	
14	
15	
16	
17	
付録	

はポート443を使用します。1~65535の任意の値を指定できます。

サポートされているコマンドレットは、「Get-Item」「Set-Item」です。

#### ● Digest <ブール値>

クライアントがダイジェスト認証を使用することを許可します。ダイジェスト認証はHTTPとHTTPSでサポートされています。

サポートされているコマンドレットは、「Get-Item」「Set-Item」です。

#### ● Enabled <ブール値>

リスナーが有効か無効かを指定します。既定値は「True」です。

サポートされているコマンドレットは、「Get-Item」「Set-Item」です。

#### ● ExactMatch <ブール値>

「Sddl」パラメータで指定されているセキュリティ設定の使用方法を指定します。「True」に設定した場合は、Sddlのセキュリティ設定を使用して、URIで指定されているURIへのアクセス試行のみが承認されます。「False」に設定した場合は、Sddlのセキュリティ設定を使用して、URIで指定されている文字列で始まるURIへのアクセス試行が承認されます

サポートされているコマンドレットは、「Get-Item」「Set-Item」です。

#### ● FileName <ファイル名>

「ResourceURI」と「SelectorSet」パラメータで指定された管理リソースの更新に使用する入力ファイルを指定します。

サポートされているコマンドレットは、「Get-Item」「Set-Item」です。

#### ● FileName (Plugin) <ファイル名>

操作プラグインのファイル名を指定します。要求を受け取ると、このエントリに入力した環境変数がユーザーのコンテキストで展開されます。

サポートされているコマンドレットは、「Get-Item」「Set-Item」です。

#### ● HostName <ホスト名>

WS-Management (WinRM) サービスが実行されているコンピュータのホスト名を指定します。値は完全修飾ドメイン名、IPv4またはIPv6リテラル文字列、またはワイルドカード文字でなければなりません。

サポートされているコマンドレットは、「Clear-Item」「Get-Item」「Set-Item」です。

#### ● IdleTimeOut <アイドル待ち時間>

リモートシェルでユーザー操作がない場合に、リモートシェルが開いたままになる最大時間をミリ秒で指定します。指定した時間が過ぎると、リモートシェルは自動的に削除されます。0~

2147483647の任意の値を指定できます。「0」はタイムアウトが無限であることを示します。既定は「900000」(15分)です。

サポートされているコマンドレットは、「Get-Item」「Set-Item」です。

#### ● IPv4Filter <IPv4アドレス>

リスナーが使用できるIPv4アドレスを指定します。このパラメータを空白にすると、IPv4アドレスは使用できません。「\*」(ワイルドカード文字)を入力した場合は、任意のIPv4アドレスを使用できます。IP範囲のリストを入力した場合は、指定した範囲のIPアドレスを使用できます。複数の範囲は「.」(コンマ)で区切り、それぞれの範囲をドット区切りの10進数形式のIPv4アドレスのペアとして「-」(ハイフン)で区切って指定します。

サポートされているコマンドレットは、「Clear-Item」「Get-Item」「Set-Item」です。

#### ● IPv6Filter <IPv6アドレス>

リスナーが使用できるIPv6アドレスを指定します。このパラメータを空白にすると、IPv6アドレスは使用できません。「\*」(ワイルドカード文字)を入力した場合は、任意のIPv6アドレスを使用できます。IP範囲のリストを入力した場合は、指定した範囲のIPアドレスを使用できます。複数の範囲は「.」(コンマ)で区切り、それぞれの範囲をドット区切りの16進数形式のIPv6アドレスのペアとして「-」(ハイフン)で区切って指定します。

サポートされているコマンドレットは、「Clear-Item」「Get-Item」「Set-Item」です。

#### ● Issuer <証明機関名>

証明書を発行した証明機関の名前を指定します。

サポートされているコマンドレットは、「Get-Item」「Set-Item」です。

#### ● Kerberos <Boolean>

クライアントがKerberos認証を使用することを許可します。「Kerberos」認証は、クライアントとサーバーがKerberos証明書を使用して相互に認証するスキームです。

サポートされているコマンドレットは、「Get-Item」「Set-Item」です。

#### ● ListeningOn <IPアドレス>

サービスが実際にリッスンするIPアドレスまたはすべてのIPアドレスを指定します。IPアドレスの値は、IPv4のドット区切りの10進数表記かIPv6のコロン区切りの16進数表記で指定します。

サポートされているコマンドレットは、「Get-Item」です。

#### ● MaxBatchItems <バッチアイテム最大数>

Pull応答で使用できる要素の最大数を指定します。1~4294967295の任意の値を指定できます。

□ 1	PowerShellの基礎知識
□ 2	
□ 3	
□ 4	
□ 5	
□ 6	
□ 7	
□ 8	
□ 9	
□ 10	
□ 11	
□ 12	
□ 13	
□ 14	
□ 15	
□ 16	
□ 17	
付録	

サポートされているコマンドレットは、「Get-Item」「Set-Item」です。

- **MaxConcurrentUsers <最大同時接続ユーザー数>**

リモートシェルから同じコンピュータに同時にリモート操作を実行できるユーザーの最大数を指定します。指定した制限を超えた場合は、新しいシェル接続が拒否されます。1~100の任意の値を指定できます。

サポートされているコマンドレットは、「Get-Item」「Set-Item」です。

- **MaxEnvelopeSizekb <SOAPデータ最大サイズ>**

最大SOAPデータをキロバイト単位で指定します。32~4294967295の任意の値を指定できます。

サポートされているコマンドレットは、「Get-Item」「Set-Item」です。

- **MaxMemoryPerShellMB <メモリ最大容量>**

アクティブなリモートシェルとそのすべての子プロセスごとに割り当てるメモリの最大合計容量を指定します。0~2147483647の任意の値を指定できます。「0」は、メモリを割り当てるリモート操作の機能が、使用可能な仮想メモリによってのみ制限されることを示します。既定値は「0」です。

サポートされているコマンドレットは、「Get-Item」「Set-Item」です。

- **MaxProcessesPerShell <プロセス最大数>**

任意のシェル操作を開始できるプロセスの最大数を指定します。0~2147483647の任意の数値を指定できます。「0」はプロセス数に制限がないことを示します。既定では、シェルごとにプロセスは5つに制限されます。

サポートされているコマンドレットは、「Get-Item」「Set-Item」です。

- **MaxShellsPerUser <シェルの最大数>**

同じシステムでユーザーがリモートから同時に開けるシェルの最大数を指定します。このポリシー設定を有効にすると、指定した限度を超えた場合に、ユーザーは新しいリモートシェルを開けなくなります。このポリシー設定を無効にした場合や構成していない場合、既定では、ユーザーごとのリモートシェル数の制限は「2」に設定されます。0~2147483647の任意の数値を指定できます。「0」はシェル数に制限がないことを示します。

サポートされているコマンドレットは、「Get-Item」「Set-Item」です。

- **MaxTimeoutMs <最大タイムアウト時間>**

Pull要求以外のすべての要求に使用できる最大タイムアウトをミリ秒単位で指定します。500~4294967295の任意の数値を指定できます。

サポートされているコマンドレットは、「Get-Item」「Set-Item」です。

□1	CHAPTER
□2	PowerShellの基礎知識
□3	
□4	
□5	
□6	
□7	
□8	
□9	
10	
11	
12	
13	
14	
15	
16	
17	
	付録

### ● Name <セッションの表示名>

WS-Managementセッションの表示名を指定します。「Get-PSSession」や「Enter-PSSession」などの他のコマンドレットを使用する場合は、この名前を使用してセッションを参照できます。この名前は、コンピュータまたは現在のセッションで一意でなくともかまいません。

サポートされているコマンドレットは、「Get-Item」「Set-Item」です。

### ● Name (Plugin) <プラグインの表示名>

プラグインに使用する表示名を指定します。プラグインからエラーが返された場合、表示名はエラーXMLに挿入され、クライアントアプリケーションに返されます。この名前はロケールに固有ではありません。

サポートされているコマンドレットは、「Get-Item」「Set-Item」です。

### ● Negotiate <プール値>

クライアントがネゴシエート認証を使用することを許可します。ネゴシエート認証は、クライアントがサーバーに認証要求を送るスキームです。サーバーは、Kerberosプロトコルを使用するかNTLMを使用するかを決定します。Kerberosプロトコルはドメインのアカウント、NTLMはローカルコンピュータのアカウントを認証する場合に選択されます。ドメインユーザーの場合、ユーザー名は「domain\\$user\_name」の形式で指定する必要があります。サーバーコンピュータのローカルユーザーの場合、ユーザー名は「server\_name\\$user\_name」の形式で指定する必要があります。

サポートされているコマンドレットは、「Get-Item」「Set-Item」です。

### ● NetworkDelayMs <遅延待機時間>

ネットワーク遅延時間に対応するためにクライアントコンピュータが待機する余分な時間をミリ秒単位で指定します。500～4294967295の任意の値を指定できます。

サポートされているコマンドレットは、「Get-Item」「Set-Item」です。

### ● Password <パスワード文字列>

ローカルアカウントまたはドメインアカウントのパスワードを指定します。NULLにすることはできません。クライアントコンピュータは、コンピュータでシェルを作成するときに使用する資格情報を指定できます。ドメインユーザーの場合、ユーザー名は「domain\\$user\_name」の形式で指定する必要があります。サーバーコンピュータのローカルユーザーの場合、ユーザー名は「server\_name\\$user\_name」の形式で指定する必要があります。

サポートされているコマンドレットは、「Get-Item」「Set-Item」です。

### ● Plugin <プラグイン>

WS-Managementの機能を拡張するプラグインを指定します。

サポートされているコマンドレットは、「New-Item」「Remove-Item」です。

## ● Port &lt;ポート番号&gt;

このリスナーを作成するTCPポートを指定します。1~65535の任意の値を指定できます。

サポートされているコマンドレットは、「Get-Item」「Set-Item」です。

## ● Resource &lt;リソース&gt;

管理操作の明確な種類または値を表すエンドポイントを指定します。サービスは1つまたは複数のリソースを公開し、リソースによっては複数のインスタンスを所有できます。管理リソースはWMIクラスまたはデータベーステーブルに似ており、インスタンスはクラスのインスタンスまたはテーブル内の行に似ています。たとえば、「Win32\_LogicalDisk」クラスはリソースを表します。「Win32\_LogicalDisk="C:\\$"」はリソースの特定のインスタンスです。

サポートされているコマンドレットは、「Get-Item」「Set-Item」です。

## ● ResourceURI &lt;リソースURI&gt;

ディスク、プロセスなど、コンピュータのリソースの種類を識別するURIを指定します。

サポートされているコマンドレットは、「Get-Item」です。

## ● RootSDDL &lt;ルートSDDL&gt;

アクセス制御エントリのセキュリティ記述子定義言語(SDDL)を指定します。これは、指定したリソースURIへのアクセス承認に使用されるセキュリティ設定を識別します。

サポートされているコマンドレットは、「Get-Item」「Set-Item」です。

## ● SDKVersion &lt;バージョン&gt;

WS-ManagementプラグインSDKのバージョンを指定します。有効な値は「1」のみです。

サポートされているコマンドレットは、「Get-Item」「Set-Item」です。

## ● Shell &lt;プロセス文字列&gt;

カスタムシェルのプロセス文字列を指定します。環境変数を指定できます。

サポートされているコマンドレットは、「Get-Item」「Set-Item」です。

## ● ShellTimeout &lt;タイムアウト時間&gt;

非アクティブであったためシェルがタイムアウトになるまでの時間を指定します。タイムアウト値をミリ秒単位で指定します。

サポートされているコマンドレットは、「Get-Item」「Set-Item」です。

## ● Subject &lt;エンティティ&gt;

証明書によって識別されるエンティティを指定します。

サポートされているコマンドレットは、「Get-Item」です。

□1	CHAPTER
□2	PowerShellの基礎知識
□3	
□4	
□5	
□6	
□7	
□8	
□9	
10	
11	
12	
13	
14	
15	
16	
17	
付録	

### ● SupportsOptions <ブール値>

プラグインがオプションの使用をサポートしているかどうかを指定します。これらは要求メッセージの「wsman:OptionSet」ヘッダー内で渡されます。

サポートされているコマンドレットは、「Get-Item」「Set-Item」です。

### ● Transport <トランSPORT>

「WS-Management」プロトコルの要求や応答の送受信に使用するトランSPORTを指定します。この値は、「HTTP」または「HTTPS」でなければなりません。「Transport」の値はリスナーの作成時に設定されます。

サポートされているコマンドレットは、「Get-Item」です。

### ● TrustedHosts <ホスト名のリスト>

信頼されているネットワーク接続でローカルコンピュータに接続しているリモートコンピュータ(ホスト)の一覧を指定します。HTTPを介した基本認証など、クライアントがサービスに認証できない認証スキームとトランSPORTを使用している場合に、この一覧に指定されているコンピュータに要求を送信できます。ホスト名にはドメインネームシステム(DNS)名かIPアドレスを指定できます。空白を指定した場合は、どのホストも信頼しません。「\*」(アスタリスク)を指定した場合は、すべてのホストを信頼します。

サポートされているコマンドレットは、「Clear-Item」「Get-Item」「Set-Item」です。

### ● URLPrefix <URLプレフィックス>

HTTPまたはHTTPS要求を受け入れるURLのプレフィックスを指定します。これは、「a-z」「A-Z」「9-0」「\_」(アンダースコア)、および「/」(スラッシュ)の文字のみを含む文字列です。文字列の先頭と末尾に「/」を使うことはできません。たとえば、コンピュータ名が「SamplePc」の場合、WS-Managementクライアントは送信先アドレスに「http://SamplePc/URLPrefix」と指定します。

サポートされているコマンドレットは、「Clear-Item」「Get-Item」「Set-Item」です。

### ● UserName <String>

ローカルアカウントまたはドメインアカウントのユーザー名を指定します。NULLの値は使用できません。クライアントは、コンピュータでシェルを作成するときに使用する資格情報を指定できます。ドメインアカウントの場合は、ユーザー名は「domain\\$user\_name」の形式で、サーバーコンピュータのローカルアカウントの場合、ユーザー名は「server\_name\\$user\_name」の形式で指定する必要があります。

サポートされているコマンドレットは、「Get-Item」「Set-Item」です。

- **Value <初期化パラメータ>**

初期化パラメータを指定します。これは、構成オプションの指定に使用されるプラグインに固有の値です。

サポートされているコマンドレットは、「Get-Item」「Set-Item」です。

- **XMLRenderingType <XMLレンダリング形式>**

「WSMAN\_DATA」オブジェクトを使用してXMLがプラグインに渡される形式を指定します。

値	説明
Text	入力XMLデータは「WSMAN_DATA_TYPE_TEXT」構造体に格納される。この構造体は、XMLをPCWSTRのメモリバッファーとして表す
XMLReader	入力XMLデータは「WSMAN_DATA_TYPE_WS_XML_READER」構造体に格納される。この構造体は、XMLを「WebServices.h」ヘッダーファイルで定義される「XmlReader」オブジェクトとして表す

サポートされているコマンドレットは、「Get-Item」「Set-Item」です。

- **xmlns <XML名前空間>**

XML名前空間を指定します。

サポートされているコマンドレットは、「Get-Item」「Set-Item」です。

- **lang <言語>**

言語（「ja」「en」など）、または言語と地域（「ja-JP」「en-US」など）を表す文字列を指定します。

サポートされているコマンドレットは、「Get-Item」「Set-Item」です。

- **HTTP <ポート番号>**

HTTPが使用されるときにクライアントが使用するポートを指定します。既定では、HTTPはポート80を使用します。1～65535の任意の値を指定できます。

サポートされているコマンドレットは、「Get-Item」「Set-Item」です。

- **HTTPS <ポート番号>**

HTTPSが使用されるときにクライアントが使用するポートを指定します。既定では、HTTPSはポート443を使用します。1～65535の任意の値を指定できます。

サポートされているコマンドレットは、「Get-Item」「Set-Item」です。

# リモート機能について

## III PowerShellのリモート機能

PowerShellでは、一時接続または固定接続を使用して、ネットワークに接続されている別のコンピュータ上でコマンド(リモートコマンド)を実行することができます。リモート処理には、WS-Management (WinRM) が使用されます。

WS-Managementは、さまざまなベンダーのハードウェアとオペレーティングシステムの相互運用を可能にする、SOAPをベースとするファイアウォール互換の標準通信プロトコルです。Microsoftが実装したWS-Managementは、WinRM(Windows Remote Management)と呼ばれます。

WinRMのリモート機能を使用してリモートコマンドを実行するには、ローカルコンピュータ(クライアントコンピュータ)およびリモートコンピュータ(ホストコンピュータ)をリモート処理用に構成する必要があります(514ページ参照)。

リモート処理用に構成されていない場合、次のコマンドレットを実行することができません。

コマンドレット	解説
New-PSWorkflowSession	506ページ
Invoke-AsWorkflow	510ページ
New-PSSession	539ページ
Get-PSSession	541ページ
Disconnect-WSMan	544ページ
Connect-PSSession	546ページ
Enter-PSSession	549ページ
Receive-PSSession	555ページ
Remove-PSSession	558ページ
Invoke-Command	565ページ
Connect-WSMan	570ページ
Get-WSManInstance	573ページ
New-WSManInstance	575ページ
Set-WSManInstance	580ページ
Test-WSMan	592ページ

## III 「ComputerName」パラメータを持つコマンドレットでのデータの取得

一部のコマンドレットには、リモートコンピュータからオブジェクトを取得するための「ComputerName」パラメータがあります。これらのコマンドレットは、コンピュータをリモート処理用に構成することなく使用できます。ただし、無条件で使用できるのは、「Get-Service」と「Set-Service」コマンドレットの2つです。その他のコマンドレットは、ファイアウォールの受信規則や依存するサービスの設定を行う必要があります。

### ▶ ファイアウォール

ファイアウォール(Windowsファイアウォール)によってブロックされるコマンドレットは、次のとおりです。

□ 1	PowerShellの基礎知識
□ 2	
□ 3	
□ 4	
□ 5	
□ 6	
□ 7	
□ 8	
□ 9	
□ 10	
□ 11	
□ 12	
□ 13	
□ 14	
□ 15	
□ 16	
□ 17	
付録	

コマンドレット	受信規則	解説
Get-WinEvent	リモートイベントログ監視(RPC)	389ページ
Get-HotFix	Windows Management Instrumentation (WMI受信)	402ページ
Restart-Computer	Windows Management Instrumentation (WMI受信)	404ページ
Stop-Computer	Windows Management Instrumentation (WMI受信)	405ページ
Rename-Computer	Windows Management Instrumentation (WMI受信)	459ページ
Get-WmiObject	Windows Management Instrumentation (WMI受信)	593ページ
Remove-WmiObject	Windows Management Instrumentation (WMI受信)	598ページ
Invoke-WmiMethod	Windows Management Instrumentation (WMI受信)	600ページ
Register-WmiEvent	Windows Management Instrumentation (WMI受信)	616ページ

これらのコマンドレットをリモートコンピュータに対して実行できるようにするには、リモートコンピュータ側のファイアウォールの設定で、表中の「受信規則」の項目を有効にします。

#### ▶ リモートレジストリサービス

リモートレジストリサービス(Remote Registry)に依存するコマンドレットは、次のとおりです。

コマンドレット	解説
New-EventLog	376ページ
Write-EventLog	378ページ
Get-EventLog	380ページ
Show-EventLog	382ページ
Limit-EventLog	384ページ
Clear-EventLog	385ページ
Remove-EventLog	387ページ
Get-Process	410ページ
Get-Counter	429ページ

これらのコマンドレットをリモートコンピュータに対して実行できるようにするには、リモートコンピュータ側でリモートレジストリサービスを起動します。リモートレジストリサービスを起動するには、次のように操作します。

- ① 次のように入力して、リモートレジストリサービスのスタートアップモードを取得します。

```
PS C:\work> Get-WmiObject win32_service -filter "name='RemoteRegistry'"
```

```
ExitCode : 1077
Name      : RemoteRegistry
ProcessId : 0
StartMode : Disabled ←スタートアップモード
State     : Stopped
Status    : OK
```

- ② スタートアップモードが「Manual」なら操作③に進みます。スタートアップモードが「Disabled」なら次のように入力して、「Manual」に設定します。

```
PS C:\work> Set-Service -Name RemoteRegistry -StartupType Manual
```

- ③ 次のように入力して、リモートレジストリサービスを起動します。

```
PS C:\work> Start-Service -Name RemoteRegistry
```

CHAPTER	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	付録
PowerShellの基礎知識																		

# 自動変数とユーザー設定変数について

## III PowerShellの自動変数

自動変数とは、PowerShellの状態情報を格納する変数です。自動変数は、PowerShellによって作成され、保持されます。自動変数の一覧を次の表に示します。

変数	説明
\$\$	セッションが受け取った最後の行にある最後のトークンを格納
\$?	最後の演算が成功した場合は「True」、失敗した場合は「False」を格納
\$^	セッションが受け取った最後の行にある最初のトークンを格納
\$_	パイプラインオブジェクトの現在のオブジェクトを格納
\$Args	宣言されていないパラメータの配列や、関数、スクリプト、またはスクリプトブロックに渡されるパラメータ値を格納
\$ConsoleFileName	セッションで最後に使用されたコンソールファイル(.psc1)のパスを格納
\$Error	最新のエラーを表すエラーオブジェクトの配列を格納
\$Event	処理中のイベントを表す「PSEEventArgs」オブジェクトを格納
\$EventSubscriber	処理中のイベントのサブスクリーバーを表す「PSEventSubscriber」オブジェクトを格納
\$ExecutionContext	PowerShellホストの実行コンテキストを表す「EngineIntrinsics」オブジェクトを格納
\$False	「False」を格納
\$ForEach	「ForEach-Object」ループの列挙子を格納
\$HOME	ユーザーのホームディレクトリの完全なパスを格納
\$Host	PowerShellの現在のホストアプリケーションを表すオブジェクトを格納
\$Input	関数に渡される入力を格納する列挙子
\$LastExitCode	最後に実行されたWindowsベースのプログラムの終了コードを格納
\$Matches	「-match」および「-not match」による演算の結果、一致した任意の文字列値のハッシュテーブルが格納される
\$MyInvocation	スクリプト、関数、スクリプトブロックなど、現在のコマンドに関する情報を含むオブジェクトを格納
\$NestedPromptLevel	現在のプロンプトレベルを格納
\$NULL	「Null」または空の値を格納
\$PID	現在のPowerShellセッションをホストしているプロセスのプロセス識別子(PID)を格納
\$Profile	現在のユーザーおよび現在のホストアプリケーションのPowerShellプロファイルの完全なパスを格納
\$PSBoundParameters	有効なパラメータとそのパラメータ現在の値のディクショナリを格納
\$PSCmdletPath (V3~)	現在実行中のスクリプトのパスを格納
\$PsCulture	オペレーティングシステムで現在使用されているカルチャの名前を格納
\$PSDebugContext	この変数は、デバッグ中にデバッグ環境に関する情報を格納
\$PSHome	PowerShellのインストールディレクトリの完全なパスを格納
\$PSItem (V3~)	変数「\$_」と同じ
\$PSScriptRoot	スクリプトモジュールが実行されているディレクトリを格納
\$PSUICulture	オペレーティングシステムで現在使用されているユーザーインターフェイス(UI)カルチャの名前を格納