Windows PowerShell w miesiąc

Wydanie III

Donald W. Jones
Jeffrey Hicks

MANNING

Helion

Tytuł oryginału: Learn Windows PowerShell in a Month of Lunches, 3rd Edition

Tłumaczenie: Grzegorz Kowalczyk

ISBN: 978-83-283-4648-2

HELION SA
ul. Kościuszki 1c, 44-100 GLIWICE
tel. 32 231 22 19, 32 230 98 63
e-mail: *helion@helion.pl*
WWW: *http://helion.pl* (księgarnia internetowa, katalog książek)

Pliki z przykładami omawianymi w książce można znaleźć pod adresem:
ftp://ftp.helion.pl/przyklady/wipom3.zip

Drogi Czytelniku!
Jeżeli chcesz ocenić tę książkę, zajrzyj pod adres
http://helion.pl/user/opinie/wipom3
Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

Printed in Poland.

Spis treści

Przedmowa 13

Podziękowania 15

O książce 17

O autorach 19

Rozdział 1. Zanim zaczniesz 21

1.1. Dlaczego nie możesz sobie pozwolić na ignorowanie powłoki PowerShell 22
1.1.1. Życie bez powłoki PowerShell 22
1.1.2. Życie z powłoką PowerShell 23
1.2. Od teraz już tylko PowerShell! 24
1.3. Czy ta książka jest dla Ciebie? 24
1.4. Jak korzystać z tej książki 25
1.4.1. Główne rozdziały 25
1.4.2. Ćwiczenia praktyczne 26
1.4.3. Przykładowe kody 26
1.4.4. Materiały uzupełniające 26
1.4.5. Dalsze pogłębianie swojej wiedzy 26
1.4.6. Dla zainteresowanych 27
1.5. Konfigurowanie środowiska testowego 27
1.6. Instalowanie Windows PowerShell 28
1.7. Kontakt z nami 30
1.8. Natychmiastowe działanie z powłoką PowerShell 30

Rozdział 2. Poznaj powłokę PowerShell 31

2.1. Wybierz swoją broń 31
2.1.1. Okno konsoli 33
2.1.2. Zintegrowane środowisko skryptowe 35

2.2. Wracamy do wpisywania poleceń z klawiatury 38
2.3. Najczęściej spotykane problemy 39
2.4. Jaka to wersja? 40
2.5. Ćwiczenia 41

Rozdział 3. Korzystanie z systemu pomocy *43*

3.1. System pomocy — jak poznawać polecenia powłoki PowerShell? 43
3.2. Aktualizowanie systemu pomocy 45
3.3. Korzystanie z systemu pomocy 47
3.4. Zastosowanie systemu pomocy do wyszukiwania poleceń 48
3.5. Interpretacja treści plików pomocy 51
3.5.1. Zestawy parametrów i parametry wspólne *51*
3.5.2. Parametry obligatoryjne i opcjonalne *52*
3.5.3. Parametry pozycyjne *53*
3.5.4. Wartości parametrów *55*
3.5.5. Wyszukiwanie przykładów poleceń *58*
3.6. Dostęp do ogólnych tematów pomocy 59
3.7. Dostęp do pomocy online 60
3.8. Ćwiczenia 60
3.9. Odpowiedzi 62

Rozdział 4. Uruchamianie poleceń *65*

4.1. Nie tworzymy skryptów, ale uruchamiamy polecenia 65
4.2. Anatomia polecenia 67
4.3. Konwencja tworzenia nazw poleceń cmdlet 68
4.4. Aliasy — skrócone nazwy poleceń 69
4.5. Tworzenie skrótów 71
4.5.1. Używanie skróconych nazw parametrów *71*
4.5.2. Używanie aliasów nazw parametrów *71*
4.5.3. Używanie parametrów pozycyjnych *72*
4.6. Trochę oszukiwania — polecenie Show-Command 73
4.7. Obsługa poleceń zewnętrznych 75
4.8. Jak radzić sobie z błędami 78
4.9. Najczęściej spotykane problemy 79
4.9.1. Wpisywanie nazw poleceń cmdlet *79*
4.9.2. Wpisywanie parametrów *80*
4.10. Ćwiczenia 80

Rozdział 5. Praca z dostawcami *83*

5.1. Czym są dostawcy? 83
5.2. Jak zorganizowany jest system plików 85
5.3. W czym system plików jest podobny do innych magazynów danych? 87
5.4. Poruszanie się w systemie plików 88
5.5. Używanie symboli wieloznacznych i dokładnych ścieżek 90

5.6. Praca z innymi dostawcami 91
5.7. Ćwiczenia 94
5.8. Co dalej? 94
5.9. Odpowiedzi 95

Rozdział 6. Potoki — łączenie poleceń 97

6.1. Łączenie poleceń ze sobą — mniej pracy dla Ciebie 97
6.2. Eksportowanie wyników działania polecenia do pliku CSV lub XML 98
6.2.1. Eksportowanie wyników działania do pliku CSV 99
6.2.2. Eksportowanie wyników działania do pliku XML 100
6.2.3. Porównywanie plików 101
6.3. Przesyłanie wyników działania polecenia do pliku lub na drukarkę 104
6.4. Konwersja na pliki w formacie HTML 105
6.5. Używanie poleceń cmdlet do modyfikowania systemu — zakończenie działania procesów i zatrzymywanie usług 106
6.6. Najczęściej spotykane problemy 108
6.7. Ćwiczenia 110
6.8. Odpowiedzi 111

Rozdział 7. Dodawanie poleceń 113

7.1. Jak jedna powłoka może zrobić wszystko 113
7.2. Powłoki przeznaczone do zarządzania danym produktem 114
7.3. Rozszerzenia — wyszukiwanie i dodawanie przystawek 115
7.4. Rozszerzenia — wyszukiwanie i dodawanie modułów 117
7.5. Konflikty poleceń i usuwanie rozszerzeń 120
7.6. Jak to wygląda w systemach operacyjnych innych niż Windows 121
7.7. Używanie nowego modułu 121
7.8. Skrypty profilów powłoki PowerShell — wstępne ładowanie rozszerzeń podczas uruchamiania powłoki 123
7.9. Pobieranie modułów z Internetu 125
7.10. Najczęściej spotykane problemy 126
7.11. Ćwiczenia 126
7.12. Odpowiedzi 127

Rozdział 8. Obiekty — dane pod inną nazwą 129

8.1. Czym są obiekty? 129
8.2. Dlaczego powłoka PowerShell używa obiektów? 131
8.3. Odkrywanie obiektów — polecenie Get-Member 133
8.4. Używanie atrybutów i właściwości obiektów 134
8.5. Używanie akcji i metod obiektu 135
8.6. Sortowanie obiektów 136
8.7. Wybieranie żądanych właściwości 137
8.8. Obiekty aż do końca 138

8.9. Najczęściej spotykane problemy 141
8.10. Ćwiczenia 141
8.11. Odpowiedzi 142

Rozdział 9. Potoki — zaglądamy nieco głębiej *143*

9.1. Potoki — większe możliwości przy mniejszej liczbie poleceń 143
9.2. Jak powłoka PowerShell przekazuje dane za pomocą potoku 144
9.3. Plan A — przekazywanie danych ByValue 145
9.4. Plan B — przekazywanie danych ByPropertyName 148
9.5. Gdy dane do siebie nie pasują — właściwości niestandardowe 153
9.6. Polecenia w nawiasach 156
9.7. Wyodrębnianie wartości z jednej właściwości 157
9.8. Ćwiczenia 163
9.9. Co dalej? 164
9.10. Odpowiedzi 165

Rozdział 10. Formatowanie wyników działania *167*

10.1. Formatowanie — upiększanie tego, co widzisz 167
10.2. Praca z formatowaniem domyślnym 168
10.3. Formatowanie tabel 171
10.4. Formatowanie list 173
10.5. Formatowanie szerokich list 174
10.6. Tworzenie niestandardowych kolumn i elementów list 175
10.7. Przesyłanie danych na wyjście: do pliku, drukarki lub na ekran 178
10.8. Jeszcze jeden typ wyjścia: GridViews 178
10.9. Najczęściej spotykane problemy 178
10.9.1. Formatuj dane na końcu *179*
10.9.2. Jeden typ obiektu jednocześnie *181*
10.10. Ćwiczenia 182
10.11. Co dalej? 183
10.12. Odpowiedzi 183

Rozdział 11. Filtrowanie i porównywanie danych *185*

11.1. Jak sprawić, aby powłoka dała Ci to, czego potrzebujesz 185
11.2. Filtrowanie z lewej 186
11.3. Korzystanie z operatorów porównania 187
11.4. Filtrowanie obiektów poza potokiem 189
11.5. Używanie iteracyjnego modelu wiersza poleceń 190
11.6. Najczęściej spotykane problemy 192
11.6.1. Filtr z lewej, proszę *193*
11.6.2. Kiedy używanie symbolu $_ jest dozwolone? *193*
11.7. Ćwiczenia 194
11.8. Co dalej? 195
11.9. Odpowiedzi 195

Rozdział 12. Trochę praktyki 197

12.1. Definiowanie zadania 197
12.2. Wyszukiwanie odpowiednich poleceń 198
12.3. Uczymy się, jak korzystać z nowych poleceń 199
12.4. Wskazówki dotyczące samodzielnej nauki 201
12.5. Ćwiczenia 201
12.6. Odpowiedzi 202

Rozdział 13. Komunikacja zdalna — jeden-do-jednego i jeden-do-wielu 203

13.1. Koncepcja komunikacji zdalnej w powłoce PowerShell 204
13.2. Usługa WinRM 206
13.3. Używanie poleceń Enter-PSSession i Exit-PSSession do połączeń zdalnych typu jeden-do-jednego 210
13.4. Zastosowanie polecenia Invoke-Command do komunikacji zdalnej typu jeden-do-wielu 213
13.5. Różnice między poleceniami zdalnymi i lokalnymi 216
13.5.1. Polecenie Invoke-Command kontra parametr -computerName 216
13.5.2. Przetwarzanie lokalne kontra zdalne 217
13.5.3. Obiekty po deserializacji 219
13.6. Ale poczekaj, jest jeszcze coś więcej 220
13.7. Opcje komunikacji zdalnej 221
13.8. Najczęściej spotykane problemy 222
13.9. Ćwiczenia 223
13.10. Co dalej? 224
13.11. Odpowiedzi 224

Rozdział 14. Zastosowanie mechanizmu WMI oraz standardu CIM 225

14.1. Podstawowe elementy WMI 226
14.2. Złe wieści na temat usługi WMI 228
14.3. Eksplorowanie WMI 230
14.4. Wybierz swoją broń: WMI lub CIM 232
14.5. Używanie polecenia Get-WmiObject 233
14.6. Używanie polecenia Get-CimInstance 237
14.7. Dokumentacja WMI 237
14.8. Najczęściej spotykane problemy 237
14.9. Ćwiczenia 238
14.10. Co dalej? 239
14.11. Odpowiedzi 239

Rozdział 15. Wielozadaniowość z zadaniami działającymi w tle 241

15.1. Uruchamianie wielu zadań powłoki PowerShell w tym samym czasie 241
15.2. Zadania synchroniczne i asynchroniczne 242
15.3. Tworzenie zadań lokalnych 243

15.4. Zapytania WMI jako zadania działające w tle 244
15.5. Komunikacja zdalna jako zadanie działające w tle 245
15.6. Pobieranie wyników działania zadania uruchomionego w tle 246
15.7. Praca z zadaniami podrzędnymi 249
15.8. Polecenia do zarządzania zadaniami 251
15.9. Zaplanowane zadania 253
15.10. Najczęściej spotykane problemy 254
15.11. Ćwiczenia 256
15.12. Odpowiedzi 256

Rozdział 16. Praca z wieloma obiektami jednocześnie 259

16.1. Automatyzacja zarządzania wieloma obiektami docelowymi 259
16.2. Preferowany sposób: polecenia „wsadowe” 260
16.3. Wykorzystanie mechanizmów CIM/WMI — wywoływanie metod 262
16.4. Plan B — wyliczanie obiektów 266
16.5. Najczęściej spotykane problemy 271
16.5.1. Który sposób postępowania jest właściwy? 271
16.5.2. Metody WMI a polecenia powłoki 273
16.5.3. Dokumentacja metod 273
16.5.4. Najczęstsze problemy z poleceniem ForEach-Object 274
16.6. Ćwiczenia 275
16.7. Odpowiedzi 275

Rozdział 17. Bezpieczeństwo wykonywania skryptów 277

17.1. Zapewnienie bezpieczeństwa powłoki PowerShell 277
17.2. Model bezpieczeństwa powłoki Windows PowerShell 278
17.3. Polityka wykonywania skryptów i podpisywanie kodu 280
17.3.1. Ustawienia polityki wykonywania skryptów 280
17.3.2. Cyfrowe podpisywanie kodu 283
17.4. Inne mechanizmy bezpieczeństwa 287
17.5. Inne luki w zabezpieczeniach? 287
17.6. Zalecenia bezpieczeństwa 288
17.7. Ćwiczenia 289

Rozdział 18. Zmienne — miejsca do przechowywania danych 291

18.1. Wprowadzenie do zmiennych 291
18.2. Przechowywanie wartości w zmiennych 292
18.3. Używanie zmiennych: zabawne sztuczki z apostrofami i cudzysłowami 294
18.4. Przechowywanie wielu obiektów w zmiennej 297
18.4.1. Praca z pojedynczymi obiektami w zmiennej 297
18.4.2. Praca z wieloma obiektami w zmiennej 298
18.4.3. Inne sposoby pracy z wieloma obiektami 299
18.4.4. Rozwijanie właściwości i metod obiektów w powłoce PowerShell v3 300
18.5. Więcej trików z cudzysłowami 301

18.6. Deklarowanie typu zmiennej 303
18.7. Polecenia do pracy ze zmiennymi 305
18.8. Dobre praktyki w pracy ze zmiennymi 306
18.9. Najczęściej spotykane problemy 306
18.10. Ćwiczenia 307
18.11. Co dalej? 307
18.12. Odpowiedzi 307

Rozdział 19. Wejście i wyjście *309*

19.1. Pobieranie i wyświetlanie informacji 309
19.2. Polecenie Read-Host 310
19.3. Polecenie Write-Host 313
19.4. Polecenie Write-Output 315
19.5. Inne sposoby wyświetlania danych 317
19.6. Ćwiczenia 318
19.7. Co dalej? 319
19.8. Odpowiedzi 319

Rozdział 20. Sesje — ułatwienie komunikacji zdalnej *321*

20.1. Ułatwienie komunikacji zdalnej z użyciem powłoki PowerShell 321
20.2. Tworzenie sesji wielokrotnego użytku i praca z nimi 322
20.3. Używanie sesji z poleceniem Enter-PSSession 323
20.4. Używanie sesji z poleceniem Invoke-Command 325
20.5. Niejawna komunikacja zdalna — importowanie sesji 327
20.6. Korzystanie z rozłączonych sesji 328
20.7. Ćwiczenia 330
20.8. Co dalej? 331
20.9. Odpowiedzi 332

Rozdział 21. Nazywasz to pisaniem skryptów? *333*

21.1. Nie tworzymy programów, ale raczej pliki wsadowe 333
21.2. Tworzenie poleceń wielokrotnego użytku 334
21.3. Parametryzowanie poleceń 336
21.4. Tworzenie sparametryzowanego skryptu 337
21.5. Dokumentowanie skryptu 339
21.6. Jeden skrypt, jeden potok 340
21.7. Szybkie spojrzenie na zasięg 344
21.8. Ćwiczenia 346
21.9. Odpowiedzi 346

Rozdział 22. Ulepszanie sparametryzowanego skryptu *349*

22.1. Skrypt bazowy 349
22.2. Zmuszamy powłokę PowerShell do ciężkiej pracy 350

22.3. Definiowanie parametrów obligatoryjnych 351
22.4. Dodawanie aliasów parametrów 354
22.5. Sprawdzanie poprawności wartości parametru 355
22.6. Wyświetlanie szczegółowych wyników działania 356
22.7. Ćwiczenia 358
22.8. Odpowiedzi 358

Rozdział 23. Zaawansowana konfiguracja komunikacji zdalnej 361

23.1. Korzystanie z innych punktów końcowych 361
23.2. Tworzenie niestandardowych punktów końcowych 362
23.2.1. Tworzenie konfiguracji sesji 363
23.2.2. Rejestrowanie sesji 364
23.3. Włączanie mechanizmu komunikacji zdalnej za pośrednictwem hostów pośrednich 367
23.4. Zaawansowane uwierzytelnianie na komputerach zdalnych 368
23.4.1. Ustawienia domyślne wzajemnego uwierzytelniania 368
23.4.2. Wzajemne uwierzytelnianie przez SSL 369
23.4.3. Wzajemne uwierzytelnianie za pośrednictwem TrustedHosts 369
23.5. Ćwiczenia 370
23.6. Odpowiedzi 371

Rozdział 24. Zastosowanie wyrażeń regularnych do parsowania plików tekstowych 373

24.1. Przeznaczenie wyrażeń regularnych 374
24.2. Podstawy składni wyrażeń regularnych 374
24.3. Używanie wyrażeń regularnych z operatorem -Match 376
24.4. Używanie wyrażeń regularnych z poleceniem Select-String 377
24.5. Ćwiczenia 379
24.6. Co dalej? 380
24.7. Odpowiedzi 381

Rozdział 25. Różne wskazówki, techniki, sztuczki i chwyty 383

25.1. Profile, podpowiedzi i kolory — dostosowywanie powłoki 383
25.1.1. Profile powłoki PowerShell 383
25.1.2. Dostosowywanie znaku zachęty powłoki 386
25.1.3. Modyfikowanie ustawień kolorów 386
25.2. Operatory: -as, -is, -replace, -join, -split, -in, -contains 388
25.2.1. Operatory -as i -is 388
25.2.2. Operator -replace 389
25.2.3. Operatory -join i -split 389
25.2.4. Operatory -contains i -in 390
25.3. Operowanie na ciągach znaków 391
25.4. Operowanie na datach 392
25.5. Operowanie na datach WMI 394

25.6. Ustawianie domyślnych wartości parametrów 395
25.7. Zastosowanie bloków skryptu 396
25.8. Więcej wskazówek, trików i technik 397

Rozdział 26. Korzystanie ze skryptów innych użytkowników *399*

26.1. Skrypt 400
26.2. Analiza wiersz po wierszu 404
26.3. Ćwiczenia 405
26.4. Odpowiedzi 407

Rozdział 27. To jeszcze nie koniec *409*

27.1. Pomysły na dalszą eksplorację powłoki PowerShell 409
27.2. „Od czego mam zacząć, kiedy przeczytałem już tę książkę?” 410
27.3. Inne zasoby, o których warto pamiętać 411

Rozdział 28. PowerShell — podręczna ściągawka *413*

28.1. Interpunkcja 413
28.2. Pliki pomocy 417
28.3. Operatory 417
28.4. Niestandardowe właściwości i kolumny 418
28.5. Pobieranie parametrów z potoku 419
28.6. Kiedy używać symbolu zastępczego $_ 420

Dodatek A. Ćwiczenia podsumowujące *423*

Skorowidz *435*

2 Poznaj powłokę PowerShell

W tym rozdziale chodzi o to, abyś zaprzyjaźnił się z powłoką PowerShell i zdecydował, którego interfejsu tej powłoki będziesz używał (tak, masz wybór!). Jeśli korzystałeś już z powłoki PowerShell, ten materiał może wydać Ci się zbędny, choć mimo to warto przejrzeć ten rozdział — możesz znaleźć tu kilka ciekawostek, które pomogą Ci w pracy z powłoką.

Niniejszy rozdział dotyczy wyłącznie powłoki PowerShell w systemie Windows. Wersje działające na innych platformach niż Windows nie mają tylu opcji i możliwości, więc jeżeli jesteś w takiej sytuacji, możesz spokojnie pominąć ten rozdział.

2.1. Wybierz swoją broń

W systemie Windows firma Microsoft oferuje dwa sposoby pracy z powłoką PowerShell (a w zasadzie nawet cztery, jeżeli jesteś bardzo wybredny). Na rysunku 2.1 pokazano stronę aplikacji ekranu *Start* z czterema ikonami powłoki PowerShell (wyróżniliśmy je obramowaniem, tak aby łatwiej można je było dostrzec).

> **WSKAZÓWKA** W starszych wersjach systemu Windows wspomniane ikony znajdują się w menu *Start*. Aby je odnaleźć, powinieneś wybrać polecenie *Wszystkie programy/Akcesoria/Windows PowerShell*. Zamiast tego możesz również wybrać z menu *Start* polecenie *Uruchom*, wpisać `powershell.exe` i nacisnąć klawisz *Enter*, co spowoduje uruchomienie konsoli powłoki PowerShell. W systemach Windows 8 i Windows Server 2012 lub nowszych przytrzymaj klawisz *Windows* na klawiaturze, naciśnij klawisz *R*, a na ekranie pojawi się okno dialogowe *Uruchom*. Aby szybko dostać się do ikon powłoki PowerShell, naciśnij i zwolnij klawisz *Windows*, a następnie w polu wyszukiwania zacznij pisać `powershell`.

Rysunek 2.1. Powłokę PowerShell możesz uruchomić na jeden z czterech sposobów

W 32-bitowym systemie operacyjnym masz tylko najwyżej dwie ikony powłoki PowerShell; w systemie 64-bitowym znajdziesz ich maksymalnie cztery. Są to:

- *Windows PowerShell* — 64-bitowa konsola tekstowa w systemie 64-bitowym; 32-bitowa konsola tekstowa w systemie 32-bitowym;
- *Windows PowerShell (x86)* — 32-bitowa konsola tekstowa w systemie 64-bitowym;
- *Windows PowerShell ISE* — 64-bitowa konsola graficzna w systemie 64-bitowym; 32-bitowa konsola graficzna w systemie 32-bitowym;
- *Windows PowerShell ISE (x86)* — 32-bitowa konsola graficzna w systemie 64-bitowym.

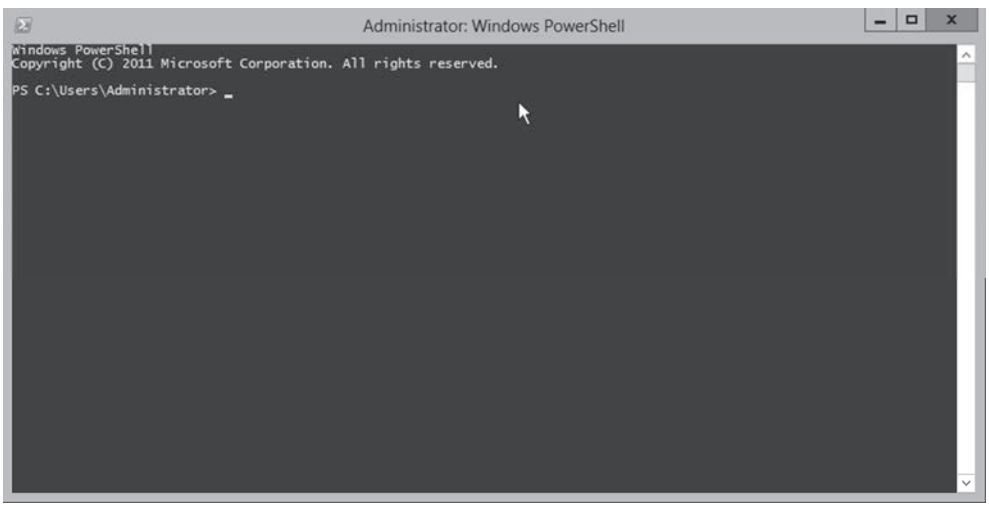
Innymi słowy, 32-bitowe systemy operacyjne mają tylko 32-bitową wersję powłoki PowerShell, a systemy 64-bitowe posiadają powłokę PowerShell zarówno w wersji 64-bitowej, jak i 32-bitowej, przy czym 32-bitowe wersje powłoki zawierają w nazwie ikony ciąg znaków *x86*. Z 32-bitowej wersji powłoki PowerShell w systemie 64-bitowym powinieneś korzystać tylko wtedy, kiedy używasz takiego rozszerzenia powłoki, dla którego wersja 64-bitowa nie jest dostępna. Obecnie firma Microsoft całkowicie postawiła na rozwój wersji 64-bitowej, a wersje 32-bitowe są utrzymywane głównie ze względu na konieczność zachowania kompatybilności wstecznej.

WSKAZÓWKA W systemach 64-bitowych można bardzo łatwo przypadkowo uruchomić niewłaściwą wersję aplikacji. Powinieneś wyrobić w sobie nawyk spoglądania na pasek tytułu okna aplikacji — jeżeli w nazwie okna znajdziesz ciąg znaków *x86*, oznacza to, że uruchamiasz aplikację 32-bitową. Rozszerzenia 64-bitowe nie będą działać w 32-bitowej wersji powłoki (a zdecydowana większość nowych rozszerzeń to wersje 64-bitowe). Dobrym, rekomendowanym przez nas rozwiązaniem jest przypięcie skrótu do wybranej wersji powłoki bezpośrednio do menu *Start*.

2.1.1. Okno konsoli

Na rysunku 2.2 przedstawiono okno konsoli, które jest miejscem, gdzie większość użytkowników po raz pierwszy styka się z powłoką PowerShell. Nieco przewrotnie rozpoczniemy tę sekcję od przedstawienia kilku argumentów przemawiających przeciwko używaniu tekstowej konsoli powłoki PowerShell:

- Konsola tekstowa nie obsługuje zestawów znaków dwubajtowych, a zatem wiele tekstów napisanych w językach innych niż angielski nie będzie wyświetlanych poprawnie.
- Operacje z użyciem schowka systemowego (kopiowanie i wklejanie) wykorzystują niestandardowe kombinacje klawiszy, do których trudno się przyzwyczaić.
- Konsola tekstowa zapewnia tylko niewielką pomoc przy wpisywaniu poleceń z klawiatury (w porównaniu z konsolą ISE, którą omówimy w dalszej części tego rozdziału), chociaż w powłoce PowerShell v5 wygląda to znacznie lepiej. Warto również zauważyć, że w systemie Windows 10 firma Microsoft wprowadziła do powłoki PowerShell wiele poprawek naprawiających niektóre z istniejących od dawna problemów, więc Twoje wrażenia mogą być nieco inne.

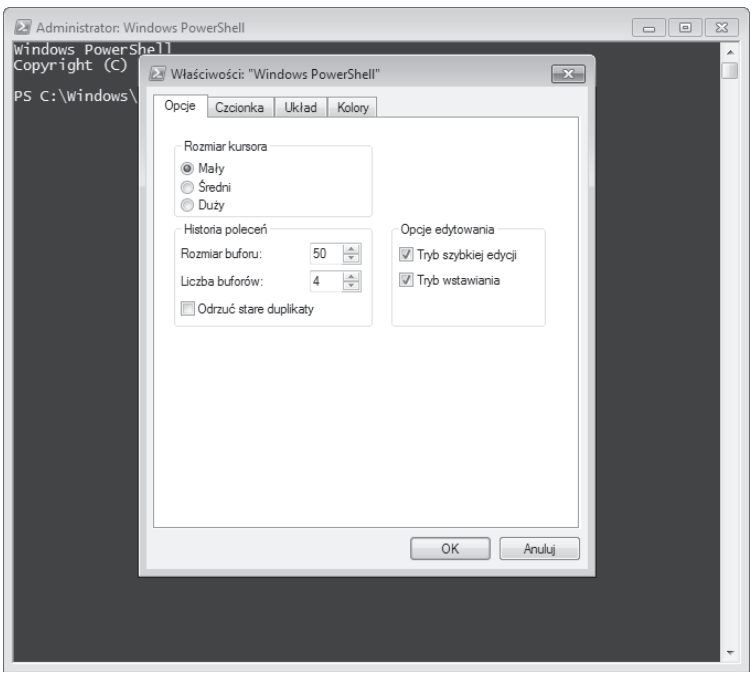


Rysunek 2.2. Standardowe okno tekstowej konsoli powłoki PowerShell — PowerShell.exe

Wymienione wyżej niedogodności nie zmieniają w niczym faktu, że konsola tekstowa jest jedynym rozwiązaniem, gdy chcesz uruchomić powłokę PowerShell na serwerach, które nie posiadają zainstalowanego graficznego interfejsu użytkownika (takich jak przykładowo serwery Server Core, Nano Server lub dowolna instancja serwera Windows Server, w której interfejs GUI został usunięty lub nie został zainstalowany). Po stronie plusów tekstowej konsoli powłoki PowerShell możemy wymienić:

- Aplikacja konsoli tekstowej ma niewielkie rozmiary, dzięki czemu szybko się ładuje i nie zużywa dużo pamięci.
- Nie wymaga żadnych innych komponentów z platformy .NET Framework oprócz tych, których potrzebuje sama powłoka PowerShell.
- W razie potrzeby możesz zmienić ustawienia kolorów na przykład na zielony tekst na czarnym tle i udawać, że pracujesz na komputerze klasy mainframe z lat 70. ubiegłego stulecia.

Jeśli zdecydujesz się na korzystanie z tekstowej konsoli powłoki PowerShell, zapoznaj się z kilkoma naszymi sugestiami dotyczącymi jej konfiguracji. Aby wprowadzić opisane niżej zmiany w ustawieniach, powinieneś kliknąć ikonę powłoki znajdującą się w lewym górnym rogu okna konsoli i z menu podręcznego wybrać polecenie *Właściwości*. Na ekranie pojawi się okno dialogowe przedstawione na rysunku 2.3. W systemie Windows 10 okno to wygląda nieco inaczej (znajdziesz tam kilka nowych opcji), ale ogólnie sposób postępowania jest taki sam.



Rysunek 2.3. Zmiana ustawień konfiguracyjnych okna konsoli

Na karcie *Opcje* możesz zwiększyć rozmiar bufora historii poleceń (opcja *Historia poleceń/Rozmiar buforu*), który pozwala konsoli zapamiętywać wpisane polecenia i umożliwia ich przywoływanie z poziomu klawiatury za pomocą strzałek w górę i w dół. Listę wcześniej wykonanych poleceń możesz także wyświetlić, naciskając klawisz *F7*.

Na karcie *Czcionka* zmień rozmiar czcionki na nieco większy niż domyślne 12 pkt. Prosimy. Naprawdę nie obchodzi nas, czy masz sokoli wzrok — po prostu zaufaj nam i zwiększ nieco rozmiar czcionki. Pracując z powłoką PowerShell, będziesz musiał często szybko odróżniać od siebie wiele podobnych znaków, takich jak ' (apostrof) czy ` (odwrócony apostrof), a mała czcionka wcale w tym nie pomaga.

Na karcie *Układ* ustaw obie opcje *Szerokość* na tę samą wartość i upewnij się, że otrzymane okno mieści się na ekranie. Jeżeli tego nie zrobisz, może się okazać, że w dolnej części okna pojawi się poziomy pasek przewijania, a niektóre, „szerokie" wyniki działania poleceń powłoki będą się pojawiały poza prawą krawędzią okna i nie będą widoczne. W czasie prowadzonych przez nas szkoleń zdarzało się, że nasi studenci spędzali sporo czasu na uporczywych próbach zmuszania do działania jakieś pozornie niedziałające polecenie, które w rzeczywistości działało jak należy, wyświetlając wyniki poza krawędzią ekranu. To może być irytujące.

Na koniec przejdź na kartę *Kolory* i postaraj się zachować umiar. Pozostaw w miarę wysoki kontrast i zachowaj łatwość czytania. Jeżeli nie podoba Ci się domyślne ustawienie kolorów (biała czcionka na niebieskim tle), to pamiętaj, że na przykład czarna czcionka na średnioszarym tle też będzie wyglądać całkiem nieźle.

Powinieneś pamiętać o jeszcze jednej sprawie: aplikacja, którą uruchamiasz w konsoli tekstowej, to nie jest powłoka PowerShell — to tylko narzędzie, za pomocą którego wchodzisz w interakcję z powłoką PowerShell. Sama aplikacja konsolowa ma swoje korzenie w połowie lat 80. ubiegłego stulecia, a co za tym idzie, jest dosyć prymitywna i nie powinieneś oczekiwać, że zrobi na Tobie ogromne wrażenie.

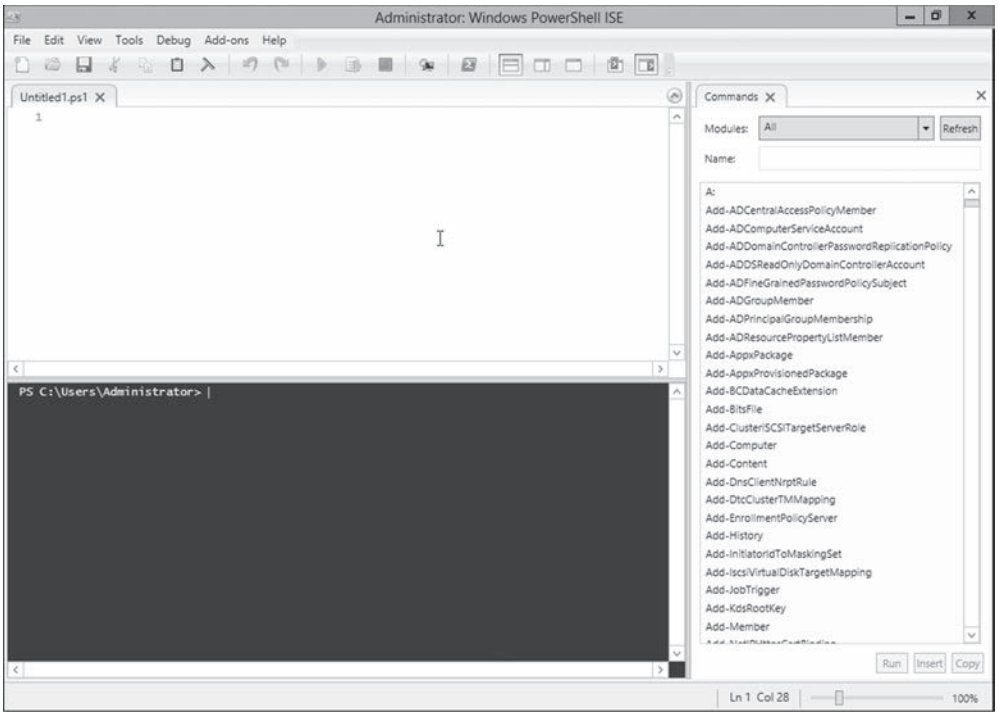
2.1.2. Zintegrowane środowisko skryptowe

Na rysunku 2.4 przedstawiono wygląd środowiska *PowerShell Integrated Scripting Environment* (ISE).

> **WSKAZÓWKA** Jeżeli przypadkowo uruchomisz standardową aplikację konsolową, to możesz uruchomić środowisko ISE, wpisując polecenie `ise` i naciskając klawisz *Enter*.

Istnieje wiele zagadnień związanych ze środowiskiem PowerShell ISE, o których warto wspomnieć, ale rozpoczniemy od przedstawienia tabeli 2.1, gdzie zamieszczamy krótkie zestawienie jego najważniejszych zalet i wad.

Zacznijmy od podstaw. Na rysunku 2.5 wyróżniono trzy główne obszary okna środowiska ISE i zaznaczono obszar paska narzędzi ISE, za pomocą którego można kontrolować te obszary.

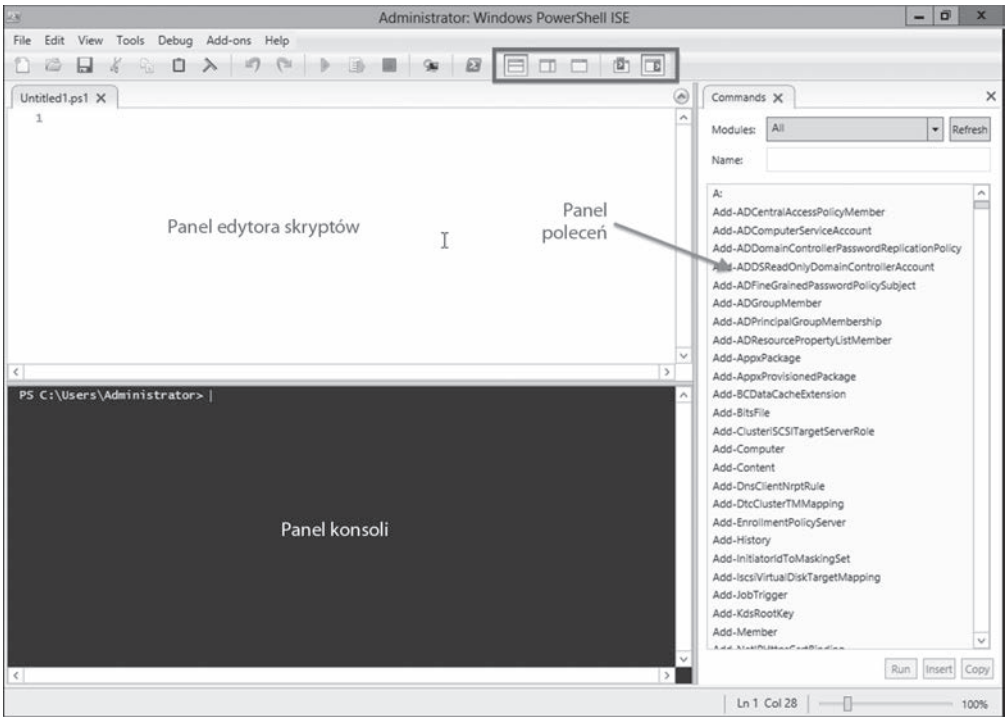


Rysunek 2.4. Środowisko PowerShell ISE (powershell_ise.exe)

Tabela 2.1. Zalety i wady środowiska PowerShell ISE

Zalety	Wady
Środowisko ISE jest znacznie ładniejsze od aplikacji konsolowej i obsługuje zestawy znaków dwubajtowych.	Wymaga obecności zainstalowanego pakietu Windows Presentation Foundation (WPF), co oznacza, że nie może działać na serwerze, na którym odinstalowano GUI (chociaż może działać w trybie GUI Minimal Server, który obsługuje aplikacje WPF).
Wspomaga tworzenie poleceń i skryptów powłoki PowerShell, o czym przekonasz się w dalszej części tego rozdziału.	Uruchamianie środowiska ISE zajmuje nieco więcej czasu, niż to ma miejsce w przypadku aplikacji konsolowej (aczkolwiek jest to różnica zaledwie kilku sekund).
Środowisko ISE wykorzystuje standardowe kombinacje klawiszy operacji kopiowania i wklejania.	Wersje wcześniejsze niż 5.0 nie obsługują mechanizmu transkrypcji.

Na rysunku 2.5 górny obszar to okienko edytora skryptów, którego w tej książce nie będziemy używać. W prawym górnym rogu tego panelu zauważysz małą niebieską strzałkę; kliknij ją, aby ukryć edytor skryptów i zmaksymalizować panel konsoli, czyli obszar, z którego będziemy korzystać. Po prawej stronie okna znajduje się panel poleceń, który możesz pozostawić otwarty lub zamknąć, klikając mały przycisk *X* w prawym górnym rogu. „Pływające" okno poleceń możesz przywołać, klikając przycisk *Show Command Window* (drugi od prawej przycisk na pasku narzędzi). Jeśli zamkniesz panel



Rysunek 2.5. Pasek narzędzi i trzy główne obszary okna środowiska ISE

poleceń i zechcesz go przywrócić, powinieneś nacisnąć przycisk *Show Command Add-on* (pierwszy od prawej na pasku narzędzi). Pierwsze trzy przyciski, które wyróżniliśmy na pasku narzędzi, sterują układem edytora skryptów i panelu konsoli. Możesz ustawić te panele jeden nad drugim, obok siebie lub otworzyć pełnoekranowy panel edytora skryptów.

W prawym dolnym rogu okna środowiska ISE znajdziesz suwak, za pomocą którego możesz zmieniać rozmiar czcionki. W menu *Tools* (narzędzia) znajdziesz polecenie *Options* (opcje), za pomocą którego można definiować niestandardowe schematy kolorów i inne ustawienia wyglądu; możesz śmiało eksperymentować z tymi ustawieniami.

ZRÓB TO SAM Przyjmujemy tutaj założenie, że podczas pracy z tą książką do tworzenia i analizowania skryptów będziesz używał wbudowanego edytora skryptów środowiska ISE. Na razie jednak ukryj okienko edytora skryptów i (jeśli chcesz) panel poleceń. Rozmiar czcionki możesz oczywiście ustawić według własnego uznania. Jeżeli domyślny schemat kolorów nie odpowiada Twoim preferencjom, również możesz go dowolnie zmodyfikować i dostosować do swoich potrzeb. Jeżeli jednak zamiast tego wolisz korzystać z aplikacji konsolowej, to oczywiście nic złego się nie wydarzy — wszystkie przykłady w książce będą nadal działać poprawnie. W kilku przypadkach będziemy jednak omawiać mechanizmy i funkcje specyficzne dla ISE, ale w takich sytuacjach zawsze Cię o tym wcześniej poinformujemy, żebyś miał szansę uruchomić środowisko ISE.

2.2. Wracamy do wpisywania poleceń z klawiatury

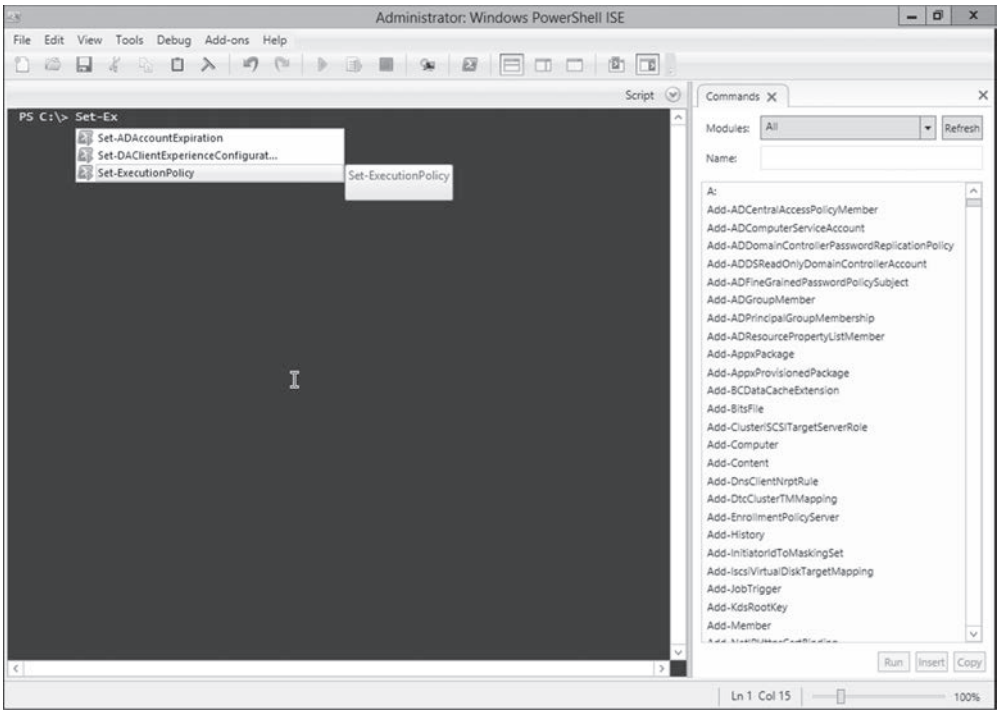
Powłoka PowerShell to interfejs wiersza poleceń, co oznacza, że będziesz wpisywać z klawiatury wiele poleceń. Podczas takiej pracy oczywiście nie sposób uniknąć prostych literówek, ale na szczęście obie aplikacje PowerShell posiadają mechanizmy wspomagające wpisywanie poleceń i minimalizowanie liczby literówek.

> **ZRÓB TO SAM** Przykładów opisanych poniżej z oczywistych przyczyn nie możemy zilustrować w książce, ale fajnie będzie je zobaczyć w działaniu. Spróbuj je wykonać w powłoce PowerShell na swoim komputerze.

Aplikacja konsolowa powłoki PowerShell posiada mechanizm dopełniania poleceń działający w czterech obszarach:

- Wpisz `Get-S` i kilka razy naciśnij klawisz *Tab*, a następnie spróbuj nacisnąć kombinację klawiszy *Shift-Tab*. Powłoka PowerShell będzie przechodziła w obie strony przez wszystkie polecenia potencjalnie pasujące do podanego wzorca. Kontynuuj naciskanie tych klawiszy aż do momentu, kiedy znajdziesz interesujące Cię polecenie.
- Wpisz `Dir`, dodaj spację, a następnie wpisz ciąg znaków `C:\` i naciśnij klawisz *Tab*. Powłoka PowerShell rozpocznie cykliczne przechodzenie do kolejnych nazw plików i podfolderów z bieżącego folderu.
- Wpisz `Set-Execu`, naciśnij klawisz *Tab*, a następnie dodaj spację i myślnik (`-`). Zacznij naciskać klawisz *Tab*, aby zobaczyć, jak powłoka PowerShell przechodzi przez kolejne dostępne parametry wybranego polecenia. Można również wpisać fragment nazwy parametru (na przykład `-E`) i nacisnąć klawisz *Tab*, aby rozpocząć dopasowywanie parametrów, których nazwy rozpoczynają się od podanego ciągu znaków. Aby wyczyścić wiersz polecenia, naciśnij klawisz *Esc*.
- Ponownie wpisz `Set-Execu` i naciśnij klawisz *Tab*. Wpisz spację, a następnie `-E` i naciśnij znów klawisz *Tab*. Wpisz kolejną spację i ponownie naciśnij klawisz *Tab*. PowerShell będzie wyświetlał kolejne dozwolone wartości tego parametru, ale działa to tylko dla parametrów, które mają predefiniowany zestaw dozwolonych wartości (taki sposób dopełniania nazywamy **wyliczaniem parametrów**). Ponownie naciśnij klawisz *Esc*, aby usunąć wpisaną komendę; nie powinieneś jeszcze teraz uruchamiać tego polecenia.

Środowisko PowerShell ISE oferuje bardzo podobny mechanizm dopełniania, nazywany IntelliSense. Mechanizm ten będzie działał we wszystkich opisanych wyżej sytuacjach z użyciem klawisza *Tab*, z tym że teraz na ekranie będzie pojawiało się fajne zgrabne podręczne menu podpowiedzi, takie jak pokazane na rysunku 2.6. Zawartość menu możesz przewijać w górę lub w dół za pomocą klawiszy strzałek. Kiedy znajdziesz żądany element, możesz go wybrać, naciskając klawisz *Tab* lub *Enter*, a następnie możesz pisać dalej.



Rysunek 2.6. Mechanizm IntelliSense w środowisku ISE odgrywa tę samą rolę co klawisz Tab w konsoli

> **OSTRZEŻENIE** To *bardzo*, *bardzo*, *bardzo*, *bardzo*, *bardzo* ważne, abyś podczas pracy z powłoką PowerShell *bardzo*, *bardzo*, *bardzo*, *bardzo* uważnie wpisywał wszystkie polecenia. W niektórych przypadkach pojedyncza zagubiona spacja, pominięty cudzysłów, apostrof czy inny znak mogą spowodować, że próba wykonania danego polecenia zakończy się niepowodzeniem. Jeżeli po uruchomieniu polecenia na ekranie pojawiają się komunikaty o wystąpieniu błędu, najpierw zawsze dwukrotnie sprawdź, co wpisałeś w wierszu polecenia.

Mechanizm IntelliSense działa w panelu konsoli ISE oraz w okienku edytora skryptów.

2.3. Najczęściej spotykane problemy

Omówimy teraz pokrótce kilka elementów, które w zupełnie niezamierzony sposób mogą Ci skomplikować życie podczas pracy z powłoką PowerShell.

- *Poziome paski przewijania w aplikacji konsolowej* — ten pozornie nieszkodliwy i bardzo przydatny mechanizm może czasami wprawić w konsternację nawet całkiem zaawansowanych użytkowników, o czym w ciągu wielu lat prowadzenia szkoleń przekonaliśmy się już niejednokrotnie. Skonfiguruj konsolę tak, aby nie miała poziomego paska przewijania u dołu okna. Nieco wcześniej w tym rozdziale opisywaliśmy już, jak możesz to zrobić.

- *32-bitowa i 64-bitowa wersja powłoki PowerShell* — powinieneś korzystać z 64-bitowej wersji systemu Windows i używać 64-bitowej wersji powłoki PowerShell (które nie mają w nazwie ciągu znaków *(x86)*). Wiemy, że dla niektórych użytkowników konieczność kupienia 64-bitowego komputera działającego pod kontrolą 64-bitowej wersji systemu Windows może nie być taka oczywista, ale jest to inwestycja, którą musisz przeprowadzić, jeżeli chcesz efektywnie korzystać z powłoki PowerShell. Większość ćwiczeń i przykładów omawianych w tej książce będzie co prawda poprawnie działać w 32-bitowej wersji powłoki PowerShell, ale szybko się przekonasz, że w środowisku produkcyjnym używanie 64-bitowej wersji tej powłoki robi poważną różnicę.
- Upewnij się, że na pasku tytułu okna powłoki PowerShell znajduje się słowo *Administrator*. Jeżeli go tam nie ma, zamknij to okno, ponownie kliknij prawym przyciskiem myszy ikonę powłoki PowerShell i z menu podręcznego wybierz polecenie *Uruchom jako administrator*. Pamiętaj, że w środowisku produkcyjnym nie zawsze będziesz mógł to zrobić, stąd w dalszej części tej książki pokażemy, jak podawać poświadczenia logowania podczas uruchamiania poleceń. Na razie musisz po prostu mieć pewność, że na pasku tytułu okna powłoki PowerShell znajduje się słowo *Administrator* — w przeciwnym razie możesz napotkać później różne problemy.

2.4. Jaka to wersja?

Sprawdzenie, której wersji powłoki PowerShell używasz, wbrew pozorom może wcale nie być takie proste, ponieważ każda kolejna wersja jest instalowana w katalogu o nazwie *1.0* (odnosi się to do wersji silnika języka skryptowego powłoki, co oznacza, że każda wersja jest zgodna z wydaniem *v1*). W powłoce PowerShell v3 i nowszych istnieje jednak łatwy sposób sprawdzenia wersji. Aby to zrobić, wpisz polecenie `$PSVersionTable` i naciśnij klawisz *Enter*:

```
PS C:\> $PSVersionTable
Name                           Value
----                           -----
PSVersion                      3.0
WSManStackVersion              3.0
SerializationVersion           1.1.0.1
CLRVersion                     4.0.30319.17379
BuildVersion                   6.2.8250.0
PSCompatibleVersions           {1.0, 2.0, 3.0}
PSRemotingProtocolVersion      2.2
```

Po wykonaniu tego polecenia na ekranie wyświetlone zostaną zarówno numery wersji poszczególnych komponentów związanych z powłoką PowerShell, jak i wersja samej powłoki. Jeżeli opisane polecenie nie działa lub jeżeli parametr `PSVersion` nie pokazuje wersji *3.0* lub nowszej, oznacza to, że używasz powłoki PowerShell w wersji, która nie jest odpowiednia dla tej książki. Informacje o tym, jak pobrać i zainstalować najnowszą wersję powłoki PowerShell znajdziesz w rozdziale 1.

ZRÓB TO SAM Nie czekaj dłużej, aby zacząć korzystać z powłoki PowerShell. Zacznij od sprawdzenia numeru wersji zainstalowanej w Twoim systemie i upewnij się, że korzystasz z wersji 3.0 lub nowszej. Jeżeli tak nie jest, zanim zaczniesz coś robić, powinieneś zainstalować powłokę PowerShell w wersji co najmniej v3.

Powłokę PowerShell v3 (i jej nowsze wersje) można zainstalować równolegle obok wersji v2. Aby później uruchomić wersję v2, możesz wykonać polecenie `PowerShell.exe -version 2.0`. Możesz tak skonfigurować powłokę PowerShell, aby automatycznie używała wersji v2, jeżeli próbujesz uruchomić coś, co nie jest kompatybilne z wersją v3 (choć to bardzo rzadki przypadek). Instalator v3 nie instaluje wersji v2, zatem powłokę v2 będziesz mógł uruchomić tylko wtedy, jeżeli została wcześniej zainstalowana jako pierwsza. Instalatory wersji v2 i v3 będą nadpisywać wersję v1 (o ile oczywiście jest już zainstalowana), ponieważ te wersje nie mogą istnieć obok siebie. Nowsze wersje, takie jak v4, mogą działać w trybie v2, ale nie mają możliwości działania w innych trybach — inaczej mówiąc, powłoka PowerShell v4 nie może działać jako v3.

WSKAZÓWKA W nowych wersjach systemu Windows domyślnie instalowana jest najnowsza wersja powłoki PowerShell, choć mogą one posiadać również silnik powłoki PowerShell w wersji v2. W razie potrzeby możesz zainstalować silnik v2, wykonując z poziomu powłoki PowerShell polecenie `Add-WindowsFeature powershell-v2`. Jeżeli opcja `powershell-v2` nie jest dostępna w Twojej wersji systemu Windows, to masz pecha, ale na szczęście w tym momencie najprawdopodobniej nie będziesz jej potrzebować.

2.5. Ćwiczenia

Ponieważ są to pierwsze ćwiczenia w tej książce, poświęcimy chwilę na opisanie, w jaki sposób powinieneś je wykonywać. W każdym zestawie ćwiczeń zamieszczamy kilka zadań, które możesz wykonać samodzielnie. Czasami dodamy również jedną lub nawet kilka podpowiedzi, tak abyś mógł pójść we właściwym kierunku.

Dajemy Ci pełną gwarancję, że wszystko, co musisz wiedzieć, aby wykonać dany zestaw ćwiczeń, znajduje się w tym samym lub w poprzednim rozdziale (a jeżeli będziesz musiał skorzystać z informacji omawianych w poprzednim rozdziale, to zazwyczaj poinformujemy Cię o tym w podpowiedziach). Nikt jednak nie twierdzi, że rozwiązania poszczególnych zadań zawsze będą proste i oczywiste. Zazwyczaj zagadnienia omawiane w danym rozdziale będą pokazywały, w jaki sposób znajdować rozwiązania różnych problemów, i przez podobny proces analizy i odkrywania będziesz musiał samodzielnie przejść podczas wykonywania ćwiczeń. Początkowo może się to wydawać nieco frustrujące, ale zmuszanie się do takiej pracy w dłuższej perspektywie z całą pewnością sprawi, że odniesiesz sukces w pracy z powłoką PowerShell. Możemy Ci to obiecać.

Pamiętaj, że na końcu każdego rozdziału znajdziesz przykładowe odpowiedzi i rozwiązania zadań. Nasze propozycje nie zawsze muszą być podobne do Twoich, a będzie się tak zapewne zdarzało coraz częściej, w miarę jak będziemy przechodzić do bardziej złożonych zagadnień. Nieraz przekonasz się, że powłoka PowerShell oferuje z pół tuzina

lub nawet więcej sposobów na wykonanie niemal każdego zadania. Pokażemy Ci metody działania, z których korzystamy najczęściej, ale jeżeli wymyślisz jakieś inne rozwiązanie, to też będzie dobrze. Każde rozwiązanie prowadzące do wykonania zadania jest poprawne.

> **UWAGA** Do wykonania tego zestawu ćwiczeń potrzebny Ci będzie dowolny komputer z zainstalowaną powłoką PowerShell w wersji 3 lub nowszej.

Zaczniemy od czegoś łatwego: chcielibyśmy tylko, abyś skonfigurował konsolę i środowisko ISE do swoich potrzeb. Wykonaj następujących pięć kroków:

1. Wybierz odpowiednie czcionki i kolory.
2. Upewnij się, że aplikacja konsoli nie ma poziomego paska przewijania u dołu. W tym rozdziale wspominaliśmy o tym co najmniej trzykrotnie, więc prawdopodobnie jest to naprawdę ważne.
3. W środowisku ISE zmaksymalizuj panel konsoli; panel poleceń możesz pozostawić lub wyłączyć według własnego uznania.
4. W obu aplikacjach wpisz pojedynczy cudzysłów (') oraz odwrócony apostrof (`) i upewnij się, że możesz je łatwo od siebie odróżnić. Na standardowych klawiaturach apostrof odwrócony znajduje się poniżej klawisza *Esc*, na tym samym klawiszu co znak tyldy (~).
5. Wpisz również znaki reprezentujące nawiasy okrągłe `()`, nawiasy kwadratowe `[]`, znaki mniejszości i większości `<>` oraz nawiasy klamrowe `{}` i upewnij się, że wybrana czcionka i jej rozmiar pozwalają na łatwe i jednoznaczne rozróżnianie wpisanych znaków. Jeżeli wygląd niektórych znaków nie jest oczywisty, zmień krój czcionki lub zwiększ jej rozmiar.

W tym rozdziale pokazywaliśmy już, jak wykonać wszystkie opisane wyżej operacje, zatem nie ma potrzeby zamieszczania żadnych odpowiedzi — sam będziesz dobrze wiedział, czy wszystko wykonałeś poprawnie.

Skorowidz

A

Active Directory, 94
akcje, 135
aktualizowanie systemu pomocy, 45
aliasy, 69
  parametrów, 354
analiza skryptu, 404
anatomia polecenia, 67
aplikacja, application, 68
  konsolowa, 38
apostrof, 294
atrybuty, 134
automatyzacja zarządzania obiektami, 259

B

bezpieczeństwo, 277
bloki skryptu, 396
błędy, 78

C

certyfikat, 284, 369
ciągi znaków, 391
CIM, 232
CLI, Command Line Interface, 43
Cmdlety
  CIM, 232
  WMI, 232
cudzysłów, 294, 301

D

daty, 392
  WMI, 394
definiowanie
  parametrów obligatoryjnych, 351
  widoku tabeli, 168
deklarowanie typu zmiennej, 303
delegowanie, 367
deserializacja, 219
dodawanie
  aliasów parametrów, 354
  modułów, 117
  przystawek, 115
dokumentacja
  metod, 273
  skryptu, 339
  WMI, 237
domyślne wartości parametrów, 395
dostawca, 83
dostęp
  do pomocy online, 60
  do statycznej metody, 313
  do tematów pomocy, 59
  do właściwości obiektu, 300
dostosowywanie znaku zachęty, 386
dyski, 85
  PSDrive, 91

E

eksportowanie wyników
  do pliku CSV, 98, 99
  do pliku XML, 98, 100
  dodawanie, 113
elementy
  nadrzędne, 344
  podrzędne, 86, 344

F

filtrowanie, 185
  obiektów, 189
  z lewej, 186, 193
foldery, 85
format HTML, 105
formatowanie
  danych, 179
  domyślne, 168
  list, 173
  tabel, 171
  wyników, 167
funkcje powłoki, 68

G

graficzne okno dialogowe, 74
graficzny interfejs użytkownika, GUI, 22

H

host pośredni, 367

I

IIS, Internet Information Server, 94
importowanie sesji, 327
instalowanie Windows PowerShell, 28
instancja, 227
IntelliSense, 39
interpunkcja, 413
ISE, Integrated Scripting Environment, 35

K

kolekcja, 130
kolor powłoki, 386
kolumny, 418
komputer
  badany, 101
  odniesienia, 101
komunikacja zdalna, 245, *patrz także* sesje, połączenie zdalne
  konfiguracja, 361
  niejawna, 327
  opcje, 221
  z wykorzystaniem hosta pośredniego, 367
komunikat o błędzie, 78
konfiguracja
  komunikacji zdalnej, 361
  sesji, 206, 361, 363
  środowiska testowego, 27
konflikty poleceń, 120
konsola
  graficzna, 309
  MMC, 113, 369
  tekstowa, 309

L

listy
  formatowanie, 173
  niestandardowe elementy, 175
  szerokie, 174
luki w zabezpieczeniach, 287

Ł

łączenie poleceń, 97

M

mechanizm
  CIM, 262
  IntelliSense, 39
  WMI, 225, 262
mechanizmy bezpieczeństwa, 287

metody, 130, 135
dokumentacja, 273
WMI, 273
wywoływanie, 262
minipowłoka, 115
MMC, Microsoft Management Console, 369
model
bezpieczeństwa, 278
połączeń, 210
moduł, 115, 117, 121
ActiveDirectory, 155
moduły
dodawanie, 117
pobieranie, 125
problemy, 126
używanie, 121
wyszukiwanie, 117
modyfikowanie
systemu, 106
ustawień kolorów, 386

N

nazwy skrócone poleceń, 69
niestandardowe
kolumny, 175, 418
właściwości, 418

O

obiekty, 129, 259
akcje, 135
atrybuty, 134
automatyzacja zarządzania, 259
deserializacja, 219
elementy składowe, 134
metody, 135
odkrywanie, 133
przechowywane w zmiennej, 297
sortowanie, 136
używanie, 131
właściwości, 134
wybieranie żądanych właściwości, 137
wyliczanie, 266
odkrywanie obiektów, 133
okno
konsoli, 33
środowiska ISE, 37
opcje komunikacji zdalnej, 221
operator, 417
-as, 388
-contains, 390
-in, 390
-is, 388
-join, 389
-Match, 376
-replace, 389
-split, 389
operatory porównania, 187
operowanie
na ciągach znaków, 391
na datach, 392
na datach WMI, 394

P

pakowanie skryptów, 279
parametr -computerName, 216
parametry
dodawanie aliasów, 354
domyślne wartości, 395
obligatoryjne, 52, 351
opcjonalne, 52
pozycyjne, 53, 72
sprawdzanie wartości, 355
wartości, 55
wspólne, 51
parametryzowanie poleceń, 336
parsowanie plików tekstowych, 373
pliki, 85
HTML, 105
pomocy, 51, 417
wsadowe, 333
pobieranie
informacji, 309
modułów, 125
parametrów z potoku, 419
wyników, 246
podpisywanie kodu, 280, 283
podwyrażenie, 302

polecenia, 43, 65
aliasy, 69, 71
anatomia, 67
do pracy ze zmiennymi, 305
do zarządzania zadaniami, 251
konflikty, 120
lokalne, 216
parametryzowanie, 336
potoki, 97, 144
przekazywania danych, 146, 149
składnia, 214
skrócone nazwy parametrów, 71
uruchamianie, 65
w nawiasach, 158
wielokrotnego użytku, 334
wpisywanie nazw, 79
wpisywanie parametrów, 80
wsadowe, 260
wyodrębnianie wartości, 159
wyszukiwanie, 198
zdalne, 216
zewnętrzne, 75
polecenie, command, 68
Invoke-Command, 213, 216
cmdlet, 68, 88, 106
Enter-PSSession, 210, 323
Exit-PSSession, 210
ForEach-Object, 274
Get-ADComputer, 160
Get-CimInstance, 237
Get-Help, 47
Get-Member, 133
Get-Process, 129
Get-WmiObject, 233
help, 101, 341
Invoke-Command, 213, 325
ise, 35
Man, 47
Out-GridView, 178
Read-Host, 310
Save-Help, 47
Select-String, 377
Show-Command, 73
Where-Object, 190, 192
Write-Host, 313
Write-Output, 315
polityka wykonywania skryptów, 280
połączenie zdalne
jeden-do-jednego, 210
jeden-do-wielu, 210, 213
tworzone ad hoc, 220
pomoc, *Patrz* system pomocy
porównywanie, 187
danych, 185
plików, 101
potoki, 97, 144, 340, 419
potokowe wiązanie parametrów, 146
PowerShell, 23, 31
PowerShell ISE, 36
problem
drugiego skoku, 367
wielu skoków, 367
problemy, 39, 79, 108, 141, 178, 192, 222, 237, 254, 271, 306
profile powłoki, 383
protokół
SSH, 204
SSL, 369
przechowywanie
wartości, 292
wielu obiektów, 297
przekazywanie danych
ByPropertyName, 149
ByValue, 146
przełącznik, 55
przepływ pracy, workflow, 68
przestrzeń nazw, 226
przesyłanie
danych, 178
wyników działania, 104
przetwarzanie
lokalne, 217
zdalne, 217
przystawka, 113, 115
Certyfikaty, 369
przystawki
dodawanie, 115
wyszukiwanie, 115
punkty końcowe, 206, 361, 362

R

reguła formatowania, 170
rejestrowanie sesji, 364
repozytorium WMI, 227
rodzaje rozszerzeń, 115

rozszerzenia, 115, 117
usuwanie, 120
rzutowanie, 388

S

sesje, 321
importowanie, 327
konfiguracja, 361, 363
rejestrowanie, 364
rozłączone, 328
wielokrotnego użytku, 322
składnia
bloku Param(), 353
poleceń, 214
wyrażeń regularnych, 374
skrypt, 66, 333
New-WebProject.ps1, 400
skrypty
analiza wiersz po wierszu, 404
bazowe, 349
bloki, 396
dokumentacja, 339
sparametryzowane, 337
profilu powłoki, 123, 383
zdalne, 282
sortowanie obiektów, 136
sprawdzanie wersji powłoki, 40
SQL Server, 94
symbol
zastępczy $_, 193, 420
wieloznaczny, 90
system
plików, 85, 87
pomocy, 43
aktualizowanie, 45
dostęp do ogólnych tematów, 59
online, 60
wyszukiwanie poleceń, 48

Ś

ścieżka, 90
środowisko
PowerShell ISE, 36
testowe, 27

T

tabele
formatowanie, 171
widok, 168
TrustedHosts, 369
tworzenie
konfiguracji sesji, 363
niestandardowych
elementów list, 175
kolumn, 175
punktów końcowych, 362
sesji, 322
zadań lokalnych, 243
typ wyjścia
GridViews, 178

U

uprawnienia punktu końcowego, 366
uruchamianie
poleceń, 65
powłoki, 32, 123
urzędy certyfikacji, 285
usługa
WinRM, 206
WMI, 228
usuwanie rozszerzeń, 120
uwierzytelnianie wzajemne, 368
przez SSL, 369
za pośrednictwem TrustedHosts, 369
zaawansowane, 368
używanie
modułu, 121
obiektów, 131
symbolu zastępczego, 420
zmiennych, 294

W

wartości parametrów, 55
wczesne filtrowanie, 185
wejście i wyjście, 309
wersja powłoki, 40
weryfikacja, 286
wielozadaniowość, 241

wiersz poleceń, 38
  model iteracyjny, 190
właściwości, 130, 134, 418
  niestandardowe, 155
WMI, Windows Management Instrumentation, 225
  dokumentacja, 237
  eksplorowanie, 230
  elementy, 226
  repozytorium, 227
  wywoływanie metod, 262
wstępne ładowanie rozszerzeń, 123
wyliczanie
  obiektów, 266
  parametrów, 38
wyodrębnianie wartości, 159
wyrażenia regularne, 373, 374
  polecenie Select-String, 377
  składnia, 374
  z operatorem -Match, 376
wyszukiwanie
  modułów, 117
  parametrów pozycyjnych, 54
  poleceń, 48, 58, 198
  przystawek, 115
wyświetlanie
  danych, 317
  informacji, 309
  wyników szczegółowych, 356
wywoływanie metod, 262

Z

zadania
  asynchroniczne, 242
  działające w tle, 244–246
  podrzędne, 244, 249
  synchroniczne, 242
  zaplanowane, 253
zakończenie działania procesów, 106
zalecenia bezpieczeństwa, 288
zapytania WMI, 244
zarządzanie
  masowe obiektami, 259
  produktem, 114
  zadaniami, 251
zasięg, scope, 344, 396
  globalny, 344
  prywatny, 344
  skryptu, 344
zasoby, 411
zatrzymywanie usług, 106
zdalna sesja powłoki, 204
zestawy parametrów, 51
zintegrowane środowisko skryptowe, ISE, 35
zmienne, 291
  deklarowanie typu, 303
  dobre praktyki, 306
  polecenia, 305
  przechowywanie
    wartości, 292
    wielu obiektów, 297
znak zachęty, 386

SIECI CISCO — NIEZAWODNE ROZWIĄZANIA!

Windows PowerShell jest rozbudowanym interpreterem poleceń i powłoką administracyjną, dzięki której można kontrolować i automatyzować prawie wszystkie zadania systemu Windows. PowerShell powstał w 2006 roku i od tej pory sukcesywnie się rozwija. Przyjmuje pojedyncze polecenia i pozwala na pisanie skryptów do zarządzania większością serwerów Windows, takich jak Exchange, IIS i SharePoint, a także usług internetowych, takich jak Azure i Office 365. Opanowanie tajników samej powłoki i związanego z nią języka skryptowego jest jednym z najważniejszych warunków bezproblemowego administrowania systemami pracującymi pod kontrolą Windows!

Dzięki temu innowacyjnemu samouczkowi i poświęceniu na naukę jednej godziny dziennie w ciągu miesiąca nauczysz się efektywnej pracy z powłoką PowerShell! Jest to trzecie, zaktualizowane wydanie książki, w której opisano funkcje powłoki działające w systemach Windows 7, Windows Server 2008 R2 i nowszych wersjach. Podręcznik składa się z 28 krótkich rozdziałów, a każdy z nich koncentruje się wokół paru kluczowych zagadnień. Opanowanie materiału zawartego w rozdziale powinno zająć najwyżej godzinę — w sam raz dla zabieganego specjalisty IT! Znajdziesz tu mnóstwo przydatnych wskazówek dotyczących poleceń i ich uruchamiania, potoków, pisania skryptów, komunikacji zdalnej, pracy na obiektach i zmiennych oraz korzystania z wyrażeń regularnych. Bardzo szybko odczujesz, że lepsza znajomość powłoki przekłada się na efektywniejsze administrowanie systemem!

W książce między innymi:

- Zalety i możliwości powłoki PowerShell
- Obiekty, potoki i formatowanie wyników
- Mechanizm WMI i standard CIM
- Zasady bezpieczeństwa w wykonywaniu skryptów
- Parametryzacja skryptów
- Ciekawe techniki, sztuczki i nieoczywiste rozwiązania

DONALD W. JONES jest laureatem prestiżowej nagrody Microsoft MVP i pozostaje uznanym autorytetem w zakresie pracy z powłoką Windows PowerShell. Obecnie pracuje jako dyrektor programowy dla IT Ops na platformie szkoleniowej Pluralsight.

JEFFREY HICKS od wielu lat zajmuje się automatyzacją i optymalizacją wydajności technologii serwerowych firmy Microsoft. Został uhonorowany nagrodą Microsoft MVP. Obecnie pracuje jako niezależny autor, trener i konsultant.

ISBN 978-83-283-4648-2
9 788328 346482
Cena: 77,00 zł