W01

PowerShell - Wprowadzenie

PowerShell to zaawansowana powłoka skryptowa opracowana przez Microsoft, będąca rozwinięciem klasycznego wiersza polecenia (CMD). PowerShell jest oparty na frameworku .NET, co umożliwia użytkownikom nie tylko wykonywanie poleceń, ale także tworzenie złożonych skryptów automatyzujących zarządzanie systemem operacyjnym, aplikacjami i infrastrukturą IT. wersji PowerShell Core, jest on wieloplatformowa, działająca na systemach Windows, macOS oraz Linux.

Główne cechy PowerShell:

1. **Obiektowość**: W odróżnieniu od CMD, PowerShell operuje na obiektach .NET, a nie na tekstowych strumieniach wyjściowych.
2. **Cmdlety**: PowerShell korzysta z cmdletów (czyt. „command-let"), czyli wbudowanych poleceń zaprojektowanych do wykonywania konkretnych operacji.
3. **Pipelining**: Umożliwia przekazywanie wyników jednego polecenia jako wejście do kolejnego (pipe |).
4. **Rozszerzalność**: Możliwość tworzenia własnych modułów i cmdletów.
5. **Integracja z systemem**: Umożliwia zarządzanie systemem operacyjnym, Active Directory, Azure, bazami danych i wieloma innymi technologiami.

Budowa typowej cmdlet w PowerShell

Cmdlety (command-lets) w PowerShell są kluczowymi poleceniami używanymi do wykonywania różnych operacji. Ich budowa jest logiczna i zrozumiała, co czyni je intuicyjnymi w użyciu. Typowa cmdlet składa się z **czasownika**, **rzeczownika** i opcjonalnie **prefiksu**.

Struktura cmdlet

1. Czasownik

- **Opisuje akcję**, którą wykonuje cmdlet, np. pobieranie, ustawianie, usuwanie, tworzenie.
- W PowerShell istnieje zdefiniowany zestaw standardowych czasowników, aby zachować spójność (np. Get, Set, New, Remove, Start, Stop, Test).
- Przykłady:
    - Get: Pobiera dane.
    - Set: Ustawia lub modyfikuje dane.
    - New: Tworzy nowy obiekt.
    - Remove: Usuwa obiekt.

2. Rzeczownik

- **Opisuje obiekt**, na którym operuje cmdlet, np. pliki, procesy, usługi, użytkownicy.
- Jest zawsze w liczbie pojedynczej, nawet jeśli cmdlet operuje na wielu obiektach.
- Przykłady:
    - Process: Proces systemowy.
    - Service: Usługa systemowa.
    - Item: Element w systemie plików.
    - User: Użytkownik.

Przykłady cmdlet

Cmdlet	**Czasownik**	**Rzeczownik**	**Opis**
Get-Process	Get	Process	Pobiera informacje o procesach.
Set-Service	Set	Service	Ustawia konfigurację usług.
New-Item	New	Item	Tworzy nowy element (np. plik, folder).
Remove-User	Remove	User	Usuwa użytkownika.
Start-Job	Start	Job	Uruchamia nowe zadanie w tle.

Dobre praktyki w tworzeniu cmdlet:

1. **Spójne używanie czasowników**: Zawsze korzystaj ze standardowych czasowników PowerShell (np. zamiast Fetch używaj Get).
2. **Czytelne rzeczowniki**: Opisuj dokładnie obiekt, na którym działa cmdlet.
3. **Moduły i prefiksy**: W przypadku konfliktów nazw lub tworzenia własnych modułów, dodawaj unikalny prefiks (np. MyModule\Get-Data).

Pomoc i Aliasy w PowerShell

Get-Help

Komenda Get-Help w PowerShell służy do wyświetlania wszystkich dostępnych informacji pomocy dla określonych cmdletów, funkcji, prac, aliasów itp.

Parametry:

- -Name: Nazwa cmdletu, funkcji, pracy, aliasu, skryptu, itp., dla którego chcesz uzyskać pomoc.
- -Category: Filtruje wyniki pomocy na podstawie kategorii, takich jak Cmdlet, Function, Workflow itd.
- -Examples: Wyświetla tylko przykłady użycia dla danego cmdletu.
- -Detailed: Wyświetla szczegółowe informacje, w tym opisy parametrów.
- -Full: Wyświetla wszystkie dostępne informacje.
- -Online: Otwiera stronę pomocy online dla danego cmdletu.
- **ShowWindow – wszystko w oknie graficznym**

Get-Command

Komenda Get-Command w PowerShell pozwala na uzyskanie wszystkich cmdletów, funkcji, aliasów itd. dostępnych w sesji.

Parametry:

- -Name: Nazwa cmdletu, funkcji, pracy, aliasu, itd., której dotyczy zapytanie.
- -CommandType: Filtruje wyniki na podstawie typu polecenia, np. Cmdlet, Function, Workflow, Alias itd.
- -Module: Filtruje wyniki na podstawie modułu, z którego pochodzi polecenie.
- -Verb: Filtruje wyniki na podstawie czasownika w nazwie polecenia.
- -Noun: Filtruje wyniki na podstawie rzeczownika w nazwie polecenia.

Obie te komendy są kluczowymi narzędziami dla każdego, kto chce efektywnie korzystać z PowerShell. Umożliwiają one zarówno zrozumienie dostępnych polecenia, jak i znalezienie pomocy w razie potrzeby.

1. Wyszukiwanie wszystkich dostępnych komend

```
Get-Command
```

2. Wyszukiwanie komend o konkretnej funkcji (np. komend związanych z plikami)

```
Get-Command *-Item
```

3. Korzystanie z Get-Help do otrzymania pomocy na temat konkretnej komendy (np. Get-Process)

```
Get-Help Get-Process
```

4. Wyświetlanie tylko przykładów użycia konkretnej komendy

```
Get-Help Get-Process -Examples
```

5. Wyszukiwanie komend z konkretnej kategorii (np. cmdletów)

```
Get-Command -CommandType Cmdlet
```

6. Wyszukiwanie modułów zainstalowanych w systemie

```
Get-Module -ListAvailable
```

7. Wykorzystanie Get-Help do znalezienia podobnych komend

```
Get-Help *Process*
```

8. Wyszukiwanie komend z określonego modułu

```
Get-Command -Module ActiveDirectory
```

9. Wyszukiwanie funkcji użytkownika

```
Get-Command -CommandType Function
```

Aliasy w PowerShell

W PowerShell, aliasy są skrótowymi nazwami dla poleceń cmdlet, funkcji, skryptów workflow, różnych komend itd. Użycie aliasów może znacznie przyspieszyć i ułatwić pracę w PowerShell, szczególnie jeśli jesteś zaznajomiony z innymi językami skryptowymi czy też jeśli chcesz uprościć długie nazwy poleceń.

Zasady korzystania z aliasów:

1. Alias jest tylko odwołaniem do oryginalnej komendy; nie zastępuje jej.
2. Wiele wbudowanych cmdletów w PowerShell ma już przypisane aliasy.
3. Możesz tworzyć własne aliasy za pomocą cmdlet **New-Alias**.

Znajdowanie Aliasów:

- Aby znaleźć alias dla konkretnej komendy, można użyć **Get-Alias**. Na przykład:

  Get-Alias -Definition Get-ChildItem pokaże wszystkie aliasy dla **Get-ChildItem**.

Przykłady użycia aliasów:

1. **Wbudowane aliasy**
    - **ls** jest aliasem dla **Get-ChildItem**
    - **pwd** jest aliasem dla **Get-Location**
    - **mv** jest aliasem dla **Move-Item**

Użycie aliasu ls zamiast Get-ChildItem ls # Użycie aliasu pwd zamiast Get-Location pwd

2. **Tworzenie własnych aliasów**

Tworzenie nowego aliasu dla Get-Process

```
New-Alias -Name "gp" -Value "Get-Process"
```

Użycie nowo utworzonego aliasu gp

3. **Usuwanie aliasów ?? :)**

```
Remove-Alias -Name "gp"
Remove-Item alias:gp
```