Wartości domyślne parametrów i plik profile - info

Działanie parametrów domyślnych

Konfigurowanie środowiska użytkownika

Plik profile użytkownika

Plik profile dla komputera lokalnego

Często uruchamiasz polecenia, które mają takie same parametry

Czy nie lepiej tak skonfigurować środowisko, że jeżeli nie podasz wartości jakiegoś wymaganego parametru to PowerShell będzie wiedział, że należy użyć tego parametru co zawsze. Komedy nie działają w ten sposób.

Istnieje zmienna $PSDefaultParameterValues która jest kolekcją elementów, możesz tu wpisać domyślne wartości parametrów które mają być używane, jeśli ty podczas uruchamiania komendy nie podałeś tego parametru.

Zainicjuj $PSDefaultParameterValues jako pusty słownik (hashtable).

```
$PSDefaultParameterValues = @{}
```

Dodaj wartości domyślne dla parametrów.

```
$PSDefaultParameterValues.Add("Get-WmiObject:Class", "Win32_OperatingSystem")
$PSDefaultParameterValues.Add("Get-WmiObject:ComputerName", "stacja")
```

Komenda bez przekazanych wartości parametrów

```
PS C:\Windows\system32> Get-WmiObject


SystemDirectory : C:\Windows\system32
Organization    :
BuildNumber     : 22631
RegisteredUser  : admin
SerialNumber    : 00330-80000-00000-AA316
Version         : 10.0.22631
```

```
PS C:\Windows\system32> Get-WmiObject -Class Win32_logicalDisk -ComputerName stacja


DeviceID     : C:
DriveType    : 3
ProviderName :
FreeSpace    : 1062053359616
Size         : 1098571051008
VolumeName   :
```

Wartości parametrów domyślnych można w dowolnej chwili zmienić podając je jawnie w komendzie.

Wykaz parametrów domyślnych

```
PS C:\Windows\system32> $PSDefaultParameterValues

Name                           Value
----                           -----
Get-WmiObject:ComputerName     stacja
Get-WmiObject:Class            Win32_OperatingSystem
```

Usuwanie parametru (nazwę parametru podajemy w taki sposób jak przy definiowaniu)

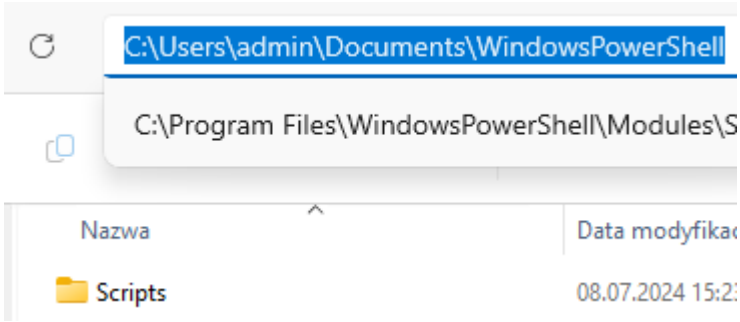
```
PS C:\Windows\system32> $PSDefaultParameterValues.Remove("Get-WmiObject:Class")

PS C:\Windows\system32> $PSDefaultParameterValues.Remove("Get-WmiObject:ComputerName")

PS C:\Windows\system32> $PSDefaultParameterValues
```

Lista parametrów domyślnych jest pusta

Jeśli zdefiniowałeś parametry dla swoich komend a potem zamknąłeś sesje PowerShell to cała praca poszła na marne. Parametry domyślne zostały utracone, istnieją tylko w ramach bieżącej sesji.

Kiedy uruchamia się środowisko PowerShell to dostajesz na wstępie zdefiniowane liczne zmienne środowiskowe, funkcje, napędy itd. Wszystkie te ustawienia są definiowane przez PowerShell automatycznie. Jeśli samodzielnie chcesz dodać własne zmienne środowiskowe, wczytać moduły, z których zawsze korzystasz to masz taką możliwość.

W momencie uruchomienia nowej sesji PowerShell automatycznie są poszukiwane i uruchamiane skrypty profilowe. Sprawdzam folder, który jest domyślmy folderem skryptów dla użytkownika admin (jeżeli by go nie było to należy utworzyć go ręcznie):



I tu niespodzianka brak pliku profile.ps1 lub Microsoft.PowerShell_profile.ps1

Musze je znaleźć. Innym problemem może być to, że nie istnieje żaden z możliwych plików profilu. Aby uzyskać rozwiązanie, użyje własnego polecenia cmdlet Test-Path programu PowerShell:

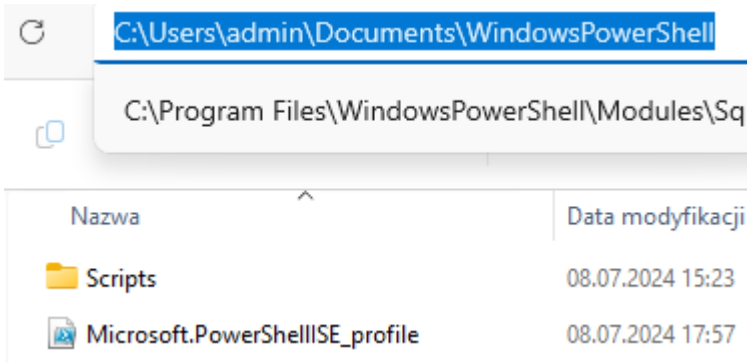
```
PS C:\WINDOWS\system32> Test-Path $Profile
False
```

$Profile jest zmienną środowiskową.

Jeśli wynik to „Fałsz”, możemy użyć PowerShell do utworzenia pliku, w ten sposób:

```
PS C:\Windows\system32> New-Item -path $profile -type file -force


    Directory: C:\Users\admin\Documents\WindowsPowerShell


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----        08.07.2024     17:57              0 Microsoft.PowerShellISE_profile.ps1
```

Sprawdzam folder, który jest domyślmy folderem skryptów dla użytkownika admin:



Tu jest plik Microsoft.PowerShell_profile.ps1

Do pliku można wpisać dowolne komendy, które mają być wykonane w momencie uruchomienia PowerShell.

Plik profilu jest skryptem, obowiązują wszystkie zasady uruchamiania skryptów w tym także te które dotyczą ExecutionPolicy

Utworzyłem przykładową zawartość pliku skryptu profilu

```
echo "Witam. Dziś $(Get-Date). Pracujesz jako $([System.Environment]::UserName)."

echo ""

echo "Twoje domyślne parametry"

$PSDefaultParameterValues = @{}

$PSDefaultParameterValues.Add("Get-WmiObject:Class", "Win32_OperatingSystem")

$PSDefaultParameterValues.Add("Get-WmiObject:ComputerName", "stacja")

$PSDefaultParameterValues

echo ""

echo ""

echo "Miłego dnia"
```

Po otwarciu nowej sesji PowerShell otrzymuje jak poniżej

```
Witam. Dziś 07/08/2024 18:05:32. Pracujesz jako admin.

Twoje domyślne parametry

Name                           Value
----                           -----
Get-WmiObject:ComputerName     stacja
Get-WmiObject:Class            Win32_OperatingSystem


Miłego dnia
```

Czy poprawnie zdefiniowałem parametr domyślny

Zostały wyświetlone informacje o systemie operacyjnym, zadziałał Win32_operatingSystem na komputerze stacja, czyli zadziałał ComputerName

Wykonuje polecenie Get-WmiObject bez podawania dodatkowych parametrów

```
PS C:\Windows\system32> Get-WmiObject


SystemDirectory : C:\Windows\system32
Organization    :
BuildNumber     : 22631
RegisteredUser  : admin
SerialNumber    : 00330-80000-00000-AA316
Version         : 10.0.22631
```

Parametry domyślne zdefiniowane teraz nie zostały utracone, istnieją w ramach każdej sesji tego użytkownika.

Jeżeli chcesz uruchamiać skrypt dla wszystkich użytkowników to należy umieścić ten skrypt w C:\Windows\System32\WindowsPowerShell\v1.0

Przydatne polecenia PowerShell dotyczące profili

Oto kilka przydatnych poleceń PowerShell, które pomogą Ci uzyskać informacje o profilach.

W wyniku uruchomienia dowolnego polecenia profilu otrzymasz lokalizację pliku profilu.

$profile

$profile.AllUsersCurrentHost

$profile.CurrentUserAllHosts

$profile.AllUsersAllHosts

```
PS C:\WINDOWS\system32> $profile
C:\Users\admin\Documents\WindowsPowerShell\Microsoft.PowerShellISE_profile.ps1

PS C:\WINDOWS\system32> $profile.AllUsersCurrentHost
C:\Windows\System32\WindowsPowerShell\v1.0\Microsoft.PowerShellISE_profile.ps1

PS C:\WINDOWS\system32> $profile.CurrentUserAllHosts
C:\Users\admin\Documents\WindowsPowerShell\profile.ps1

PS C:\WINDOWS\system32> $profile.AllUsersAllHosts
C:\Windows\System32\WindowsPowerShell\v1.0\profile.ps1
```

Użyj tego wiersza kodu, aby uzyskać lokalizacje profili dla wszystkich typów profili.

$profile | Get-Member -Type NoteProperty | Select-Object Definition

```
PS C:\WINDOWS\system32> $profile | Get-Member -Type NoteProperty | Select-Object Definition

Definition
----------
string AllUsersAllHosts=C:\Windows\System32\WindowsPowerShell\v1.0\profile.ps1
string AllUsersCurrentHost=C:\Windows\System32\WindowsPowerShell\v1.0\Microsoft.PowerShellISE_profile.ps1
string CurrentUserAllHosts=C:\Users\admin\Documents\WindowsPowerShell\profile.ps1
string CurrentUserCurrentHost=C:\Users\admin\Documents\WindowsPowerShell\Microsoft.PowerShellISE_profile.ps1
```

Jak wybrać plik profilu do użycia?

Jeśli używasz wielu aplikacji hosta, takich jak Windows PowerShell Console i ISE , umieść wszystkie dostosowania w profilu $PROFILE.CurrentUserAllHosts dla bieżącego zalogowanego użytkownika. Na przykład utworzyłem własne cmdlety zorganizowane w moduły, więc umieszczam polecenia ładowania tych modułów w tym profilu, ponieważ chcę, aby moduły były ładowane zarówno po otwarciu konsoli PowerShell, jak i ISE.

Jeśli chcesz dokonać określonego dostosowania aplikacji hosta, umieść cały kod dostosowania w profilu specyficznym dla tej aplikacji hosta. Na przykład zmieniam kolor tła czcionek PowerShell Console, aby były bardziej widoczne lub załadowałem moduły, które są dodatkami do ISE.

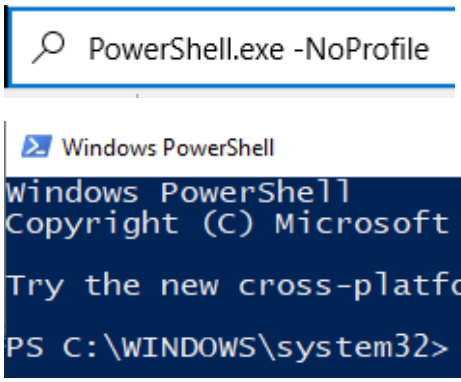
Jeśli chcesz dostosować PowerShell dla wielu użytkowników i jesteś administratorem, postępuj zgodnie z poniższymi wskazówkami:

1. Przechowuj najczęstsze dostosowania w profilu $PROFILE.AllUsersAllHosts, ponieważ jest to najszerszy zakres
2. Jeśli chcesz dostosować program Windows PowerShell dla wszystkich użytkowników, ale tylko dla aplikacji hosta. Na przykład inne dostosowanie dla konsoli Windows PowerShell i inne dostosowanie dla Windows PowerShell ISE, a następnie umieść kod w profilach $PROFILE.AllUsersCurrentHost
3. Napisz kod dla poszczególnych użytkowników w profilach specyficznych dla użytkownika. Zasadniczo oznacza to użycie profilu $PROFILE.CurrentUserAllHosts lub $PROFILE.CurrentUserCurrentHost .

Jak uruchomić konsolę lub ISE bez załadowanych profili

Otwórz okno dialogowe uruchamiania Win + R i otwórz PowerShell z opcją -NoProfile.

PowerShell.exe -NoProfile



Profile i sesje zdalne

Profile programu PowerShell nie są uruchamiane automatycznie w sesjach zdalnych, więc polecenia dodawane przez profile nie są obecne w sesji zdalnej. Ponadto zmienna automatyczna $PROFILE nie jest wypełniana w sesjach zdalnych.

Aby uruchomić profil w sesji, użyj polecenia Cmdlet Invoke-Command .

Na przykład następujące polecenie uruchamia profil „Bieżący użytkownik, bieżący host” z komputera lokalnego w sesji w $s.

Invoke-Command -Session $s -FilePath $PROFILE

Znasz zasadę działania parametrów domyślnych

Wiesz, jak konfigurować środowisko użytkownika

Znasz zastosowanie pliku profile użytkownika

Znasz zastosowanie pliku profile dla komputera lokalnego