




Windows PowerShell

Čo je Powershell a na čo je dobrý?

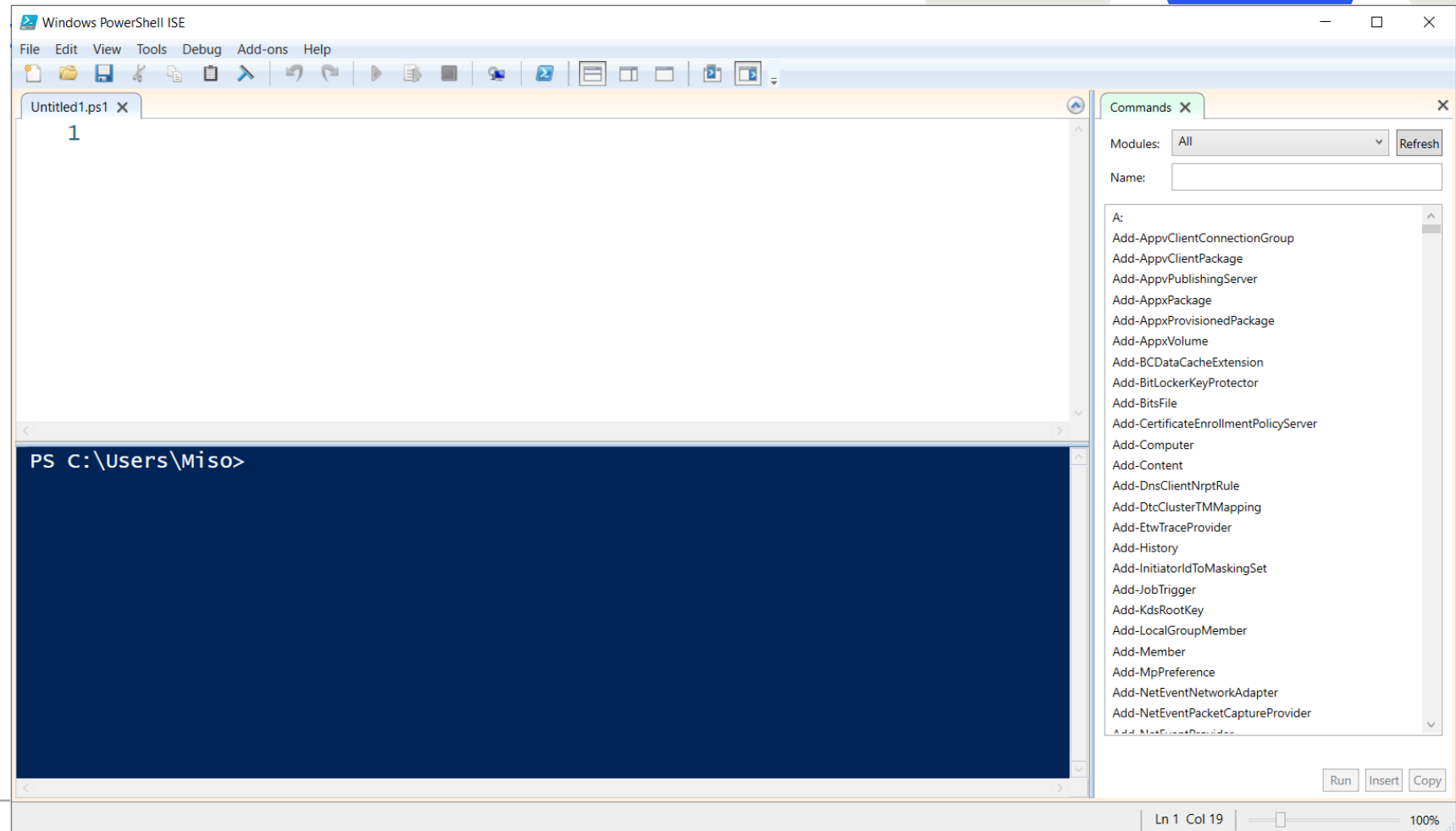
- Čo je PowerShell?
 - Objektovo-orientovaný programovací jazyk
 - Interaktívny interpreter príkazov v OS (CLI)
 - Kedy nám pomôže Powershell?
 - Pri automatizácii systémových úloh – hromadné (dávkové) spracovanie
 - Pre vytváranie manažovacích nástrojov pre často sa opakujúce procesy
-

Ako spustiť Powershell?

- Z príkazového riadku zadáním príkazu *powershell*
 - Klávesová skratka +R a zadanie príkazu *powershell*
 - Z ponuky štart výberom Powershell
 - Môže byť spustený s používateľskými alebo administrátorskými právami
-

Powershell ISE

Powershell má
vlastné vývojové
prostredie, ktoré sa
označuje skratkou
ISE (Integrated
Scripting
Environment)



Command-let (cmdlet)

- Sú príkazy vytvorené pre použitie v powershelli
 - Rôzne časti systému (napríklad roly vo WS) si vedia do powershellu doplniť/doinštalovať (tzv. registrovať) svoje vlastné cmdlety
 - Cmdlety sú zoskupené do modulov, ktoré je možné do powershellu naimportovať
 - Powershell sa snaží interpretovať všetko ako cmdlet
 - Ak sme v nejakom priečinku a máme v ňom spustiteľný súbor napr. *hra.exe*, powershell by po napísaní slova *hra* vypísal hlášku, že takýto cmdlet nepozná. V tom prípade by nám pomohlo zadať pred príkaz znaky `.\` teda napísať `.\hra`
-

Command-let (cmdlet)

- Názov cmdletu je tvorený dvojicou sloveso - podstatné meno
 - Sloveso popisuje akciu a podstatné meno popisuje zdroj, nad ktorým sa akcia má vykonať
 - Napríklad **Get-NetAdapter** vráti zoznam všetkých sieťových adaptérov v systéme
 - Všetky slovesá získame príkazom **Get-Verb**
 - Zoznam všetkých cmdletov získame zadáním príkazu **Get-Command**
 - Nápovedu k príkazu získame zadáním cmdletu **Get-Help príkaz** (napr. *Get-Help Get-Command*)
-

Pomocník (help)

- Aktuálnu verziu pomocníka (aktualizáciami sa môžu cmdlety meniť) získame zadaním príkazu ***Update-Help -Force***
 - Pomocníka vyvoláme príkazom ***Get-Help príkaz***
 - S prepínačom ***-Full*** dosiahneme výpis plného helpu
 - S prepínačom ***-Parameter názov*** dosiahneme výpis helpu k danému parametru
 - S prepínačom ***-Examples*** dosiahneme výpis príkladov použitia cmdletu
 - Zoznam príkazov, ktoré obsahujú slovo **IP** získame použitím príkazu ***Get-Command -Name *IP**** (kde * funguje ako wildcard)
-

Aliases

- Aliasy sú zástupné názvy pre cmdlety
 - Ich úlohou je skrátiť dlhé názvy cmdletov, alebo uľahčiť administrátorom, ktorí sú zvyknutí na príkazový riadok linuxu, používanie powershellu (napr. aliasy *ps*, *ls*, *cp*, *rm*, *man* ...)
 - Zoznam aliasov a ich pôvodných príkazov vypíšeme cez ***Get-Alias***
-

Objekty

- Na rozdiel od príkazového riadku (cmd) vo Windowse alebo v Linuxe, kde boli výstupom textové reťazce, sú výstupom z powershell cmdletov objekty
 - Každý objekt má viacero vlastností a každá vlastnosť má svoju hodnotu
 - Každý objekt môže mať aj svoje metódy, ktoré umožňujú prácu s týmto objektom
-

Objekty

- Zoznam vlastností a metód fungujúcich nad daným objektom môžeme získať zreťazením výstupu (cez znak | pipe) jedného príkazu s príkazom **Get-Member**
 - Napríklad **Get-Process | Get-Member** alebo **Get-ChildItem | Get-Member**
 - Tieto metódy a vlastnosti vieme ďalej využiť pri tvorbe skriptov
 - Môžete si vyskúšať vypísať len plné názvy priečinkov tak, že vyvoláte len vlastnosť *FullName* zadáním **(Get-ChildItem).FullName**
-

Ovplyvnenie formátu výstupu

- Každý príkaz má svoj vlastný formát výstupu (teda počet a tvar vlastností výstupných objektov)
 - Zmenu formátu výstupu dosiahneme zreťazením s príkazom **Format-X**, kde X môže byť najčastejšie List, Table, Wide
 - Napríklad **Get-Process | Format-List**
 - Parametre, ktoré sa majú vypisovať môžeme ovplyvniť prepínačom **-Property** (parametre-properties získame cez **Get-Member**)
 - Napríklad **Get-Process | Format-Table -Property Name,Id,StartTime**
-

Zoradenie výstupu

- Výstup cmdletu môže byť zoradený podľa toho, ako nám to vyhovuje
 - Spravíme tak zreťazením s príkazom **Sort-Object**, za ktorým uvedieme názov vlastnosti/í, podľa ktorej/ých chceme zoradovať a prípadne aj smer zoradenia (bez udania je vzostupné a-z, ak chceme naopak, zostupne z-a, zvolíme prepínač **-Descending**)
 - Napríklad zoradenie procesov zostupne (od najväčšieho po najmenšie) podľa ID procesu získame zadaním **Get-Process | Sort-Object Id -Descending**
 - Unikátne hodnoty získame prepínačom **-Unique**
-

Filtrovanie výstupu

- Pre odstránenie niektorých objektov z výstupu resp. ponechanie len niektorých častí výstupu môžeme použiť filtrovanie pomocou zreťazenia s príkazom **Where-Object**
 - Filtrovacia podmienka pozostáva z 3 častí:
 - Vlastnosť (property)
 - Porovnávací operátor
 - Hodnota pre porovnanie
-

Filtrovanie výstupu

Najčastejšie porovnávacie operátory

| | | |
|----------|-------------|--|
| -eq | rovný | vyberie len presnú zhodu s hodnotou |
| -neq | nerovný | vyberie všetko mimo zhody s hodnotou |
| -lt | menší | vyberie všetky objekty s menšou hodnotou |
| -le | menší/rovný | |
| -gt | väčší | vyberie všetky objekty s väčšou hodnotou |
| -ge | väčší/rovný | |
| -like | podľa | vyberie všetky objekty podľa masky s wildcardami |
| -notlike | nie podľa | vyberie objekty, ktoré sa nezhodujú s maskou |

Filtrovanie výstupu

Príklad použitia s jednoduchým zápisom

Odfiltruje procesy, ktoré bežali na procesore menej ako 2 sekundy:

```
Get-Process | Where-Object -Property TotalProcessorTime -lt 2
```

Odfiltruje proces, ktorého ID je rovné číslu 1000:

```
Get-Process | Where-Object -Property ID -eq 1000
```

Odfiltruje procesy, ktoré majú v názve slovo host:

```
Get-Process | Where-Object -Property ProcessName -like "*host*"
```

Filtrovanie výstupu

Príklad použitia s komplexnejším zápisom

Odfiltruje procesy, ktoré bežali na procesore menej ako 2 sekundy:

```
Get-Process | Where-Object {$_.TotalProcessorTime -lt 2}
```

Odfiltruje proces, ktorého ID je rovné číslu 1000:

```
Get-Process | Where-Object {$_.ID -eq 1000}
```

Odfiltruje procesy, ktoré majú v názve slovo host:

```
Get-Process | Where-Object {$_.ProcessName -like "*host*"}
```

Filtrovane výstup

Ďalšie príklady použitia

Skrátený zápis, kde Where-Object nahradíme zápisom |?:

```
Get-Process |? {$_.TotalProcessorTime -lt 2}
```

Spojenie podmienok logickou spojkou and (a súčasne) – procesy, ktoré bežali na procesore od 2 do 10 sekúnd (vrátane):

```
Get-Process |? {($_.TotalProceesorTime -ge 2) -and ($_.TotalProceesorTime -le 10)}
```

Spojenie podmienok logickou spojkou or (alebo) – procesy, ktoré majú v názve host alebo audio:

```
Get-Process |? {($_.ProcessName -like "*host*") -or ($_.ProcessName -like "*audio*)}
```

Ovplyvnenie počtu objektov vo výstupe

- Počet riadkov vo výstupe alebo ich unikátnosť vieme ovplyvniť cez zrežanie s príkazom **Select-Object**
 - Select-Object má prepínače **-First** a **-Last**, za ktorými uvedieme počet záznamov, ktoré sa majú zobrazíť:
 - **Get-Process | Select-Object -First 10** zobrazí prvých 10 procesov z výpisu
 - **Get-Process | Select-Object -Last 10** zobrazí posledných 10 procesov z výpisu
 - **Get-Process | Select-Object -Index 10** zobrazí desiaty proces z výpisu
 - Pre odstránenie opakujúcich sa záznamov použijeme prepínač **-Unique**
-

Príklad jednoduchého skriptu s cyklom

```
$folders = Get-Content .\folders.txt | Sort-Object  
foreach ($folder in $folders){  
    New-Item "E:\test\${folder}" -ItemType "directory"  
}
```

Tento skript vezme každý riadok zo súboru folders.txt (v aktuálnom priečinku), zoradí ich podľa abecedy a v zložke E:\test\ vytvorí pre každý riadok jednu zložku s názvom, ktorý je uvedený na tomto riadku.
