

Windows 2008 R2 PowerShell

Lumír Návrát

- COMMAND.COM (DOS, Win 9x)
- CMD.EXE (NT)
 - Základní příkazy, spouštění programů, primitivní skriptování (*.BAT, *.CMD)
- Windows Script Host (*.VBS)
 - Nemožnost interaktivního vykonávání
- PowerShell

Shelly v MS systémech

- 2003 – první zmínka
- 2006 v 1.0
 - Volitelně pro XP, 2003, Vista, 2008
- 2009 v 2.0
 - Integrována ve Win7, 2008 R2
 - Dnes volitelně i pro starší verze (XP+)
- 2012 v 3.0 součást Windows Management Framework
 - W7, W2008, W2008R2 a novější
- 2013 v 4.0
 - Spíš aktualizace, podpora Remote debugging
- 2008 Pash
 - Začátek portace pod *NIX systémy v rámci projektu MONO
- 2016 Převedeno pod Open Source

Historie PowerShellu

- Cmd-lets [Commandlets]
 - Nativní příkazy, jedná se o .NET třídy
- PowerShell skripty
 - Soubory *.PS1
- PowerShell funkce
- Spustitelné soubory
 - Kompatibilita s „CMD.EXE“ příkazy
- PowerShell není case sensitive

Příkazy PowerShellu

- Stromové struktury jsou přístupné přes providery.
 - Není omezení jen na disky, složky, soubory
- Souborový systém
 - Stejně jako jiná příkazová prostředí
 - C: D: E:
- Registry Windows
 - HKLM: HKCU:
- Systémové proměnné
 - Env:
- Proměnné
 - [scope]:jméno

Provideři PowerShellu

- Nativní příkazy (.NET třídy)
- Pevně daná struktura <verb>-<noun>
- Například:
 - Get-Help
 - Get-ChildItem
 - Start-Service
 - Format-Table
 - New-ADUser
 - ...
- Doplnování názvů pomocí TAB

Cmd-lets

- Zástupné názvy pro příkazy
- například: „DIR“ nebo „LS“
 - Jsou aliasy pro Get-ChildItem
- Je možno vytvářet vlastní
 - New-Alias

Alias

```

PS C:\> Get-Alias | `
>> Where-Object {$_.Definition -like '*-Object'} | `
>> Format-Table Definition, Name -AutoSize
>>

```

Definition	Name
Compare-Object	diff
ForEach-Object	foreach
ForEach-Object	%
Group-Object	group
Select-Object	select
Sort-Object	sort
Tee-Object	tee
Where-Object	where
Where-Object	?

```

PS C:\> gal|?{$_.Definition-like '*-ob*'}|ft n*,def* -a

```

Aliasy (about_aliases)

- Jednotná nápověda
- Get-Help
- Příklad použití: Get-Help Get-ChildItem
- Nápověda k nápovědě: Get-Help Get-Help
- 4 úrovně:
 - Základní
 - Detailní (-detailed)
 - Kompletní (-full)
 - Příklady (-examples)

Nápověda

- Předávání dat mezi příkazy pomocí „|“
- Nepředává se text, ale objekty
 - Formátování výstupu je nezávislé
- Get-Process | Sort-Object

Objektová pipeline

```
Wybrat Windows PowerShell
PS C:\> Get-Process | Format-Table
```

Handles	NPM(K)	PM(K)	WS(K)	UM(M)	CPU(s)	Id	ProcessName
46	4	1292	2908	15		1664	AESTSr64
63	8	2568	5872	72		4644	ApMsgFwd
80	8	2412	5576	70	9.98	4708	ApntEx
136	12	3352	10076	81	4.43	3168	ApntEx
75	8	1224	3936	42		1620	armsvc
130	10	15920	16004	54		4612	audiodg
851	36	16920	29844	124		2116	CcmExec
53	7	2936	7276	66	1.97	1304	conhost
47	6	1744	4996	63	0.02	4724	conhost
783	14	2788	5096	50		416	csrss
476	20	3292	102716	214		484	csrss
125	15	2572	6712	75		1708	cvpnd
177	11	3448	9792	82	0.09	4252	DCPSysMgr
198	14	3284	10400	79		2264	DCPSysMgrSvc
252	16	2552	6608	64		1860	dsAccessService
116	11	1776	5436	62		1776	dsNcService
129	16	83164	47024	256	64.80	4028	dwm

```
Windows PowerShell
PS C:\> Get-Process | Format-List
```

Id : 1664
Handles : 46
CPU :
Name : AESTSr64

Id : 4644
Handles : 63
CPU :
Name : ApMsgFwd

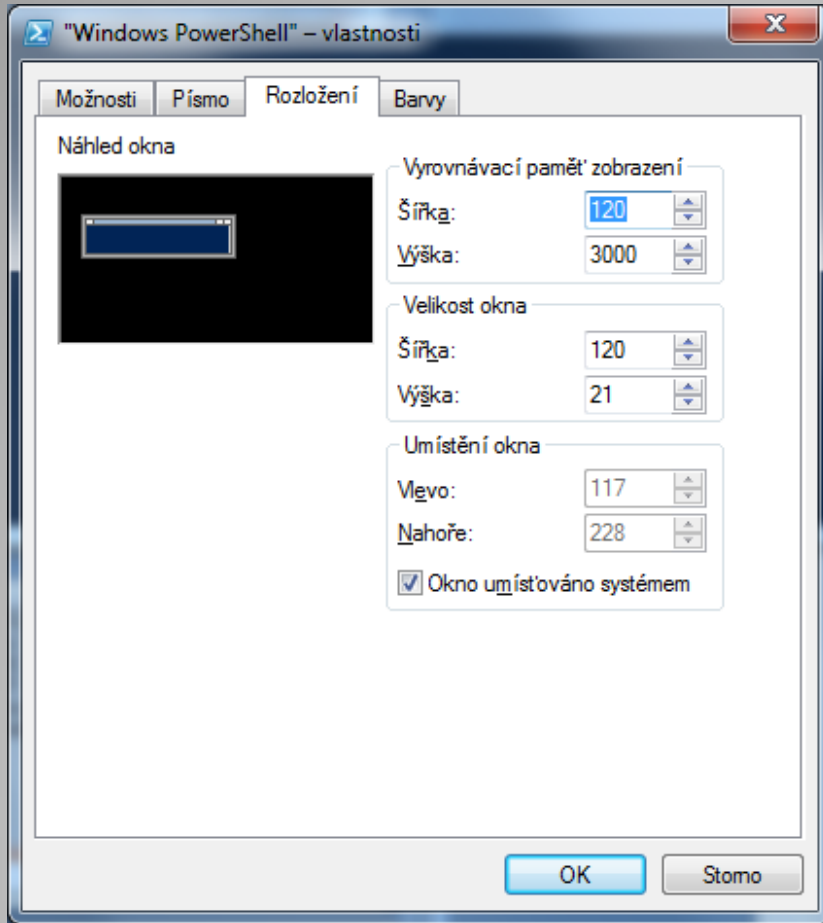
Id : 4708
Handles : 80
CPU : 10.0776646
Name : ApntEx

Id : 3168
Handles : 136
CPU : 4.4616286

Formátování výstupu

- ↑ ↓ Listování mezi použitými příkazy
- → Naposledy použitý příkaz po zn.
- F3 Naposledy použitý příkaz
- F7 Seznam použitých příkazů
- Esc Vymázání zadaného příkazu
- Tab Doplnění názvu (příkaz, objekt)

Práce s prostředím



- Nastavení paměti zobrazení, může limitovat výstup příkazu
- Další možnosti
 - Režim rychlých úprav
 - Přímé označování
 - Copy = Enter
 - Paste = pr.tl. myši

Práce s prostředím

- Syntaxe: \$název
- Nedeklarují se
- Bez typové kontroly
- Příklad použití:
 - \$cislo = 2
- \$_
 - Reprezentuje aktuální objekt v rouře
- \$args
 - Parametry příkazu

Proměnné (about_variables)

- Filtrování dle podmínky
 - Where-Object {<podmínka>}
- Operátory (about_Comparison_Operators)
 - -eq ... = -gt ... > -lt ... <
 - -ne ... <> -ge ... >= -le ... <=
 - -and -or -not
 - -like (maska) -match (reg. Výrazy)

Filtrování objektů

- Výběr objektů dle pořadí
 - Select-Object
 - -First, -Last
- Třídění objektů
 - Sort-Object
 - Sort-Object -descending

Výběr a třídění objektů

- Následují příklady

Příklady

- Get-Process
 - Vypíše běžící procesy
- Get-Process | Sort-Object WS -Descending
 - Vypíše běžící procesy seřazeny dle využití paměti
- Get-Process | Where-Object { \$_.WS -gt 20MB }
 - Vypíše běžící procesy s využitím paměti větším než 20MB
- Get-Process | Sort-Object Handles -Descending | Select-Object -first 3 | Export-Csv C:\procesy.csv
 - Zapiše do CSV souboru 3 procesy s nejvyšším počtem handles v systému

Příklady

- Get-Service
 - Vypíše služby v systému
- Get-Service | Get-Member
 - Vypíše všechny vlastnosti objektu Service
- Get-Service | Where-Object { \$_.displayname -like "*Firewall"} | Stop-Service
 - Zastaví službu, jejíž název končí na „Firewall“

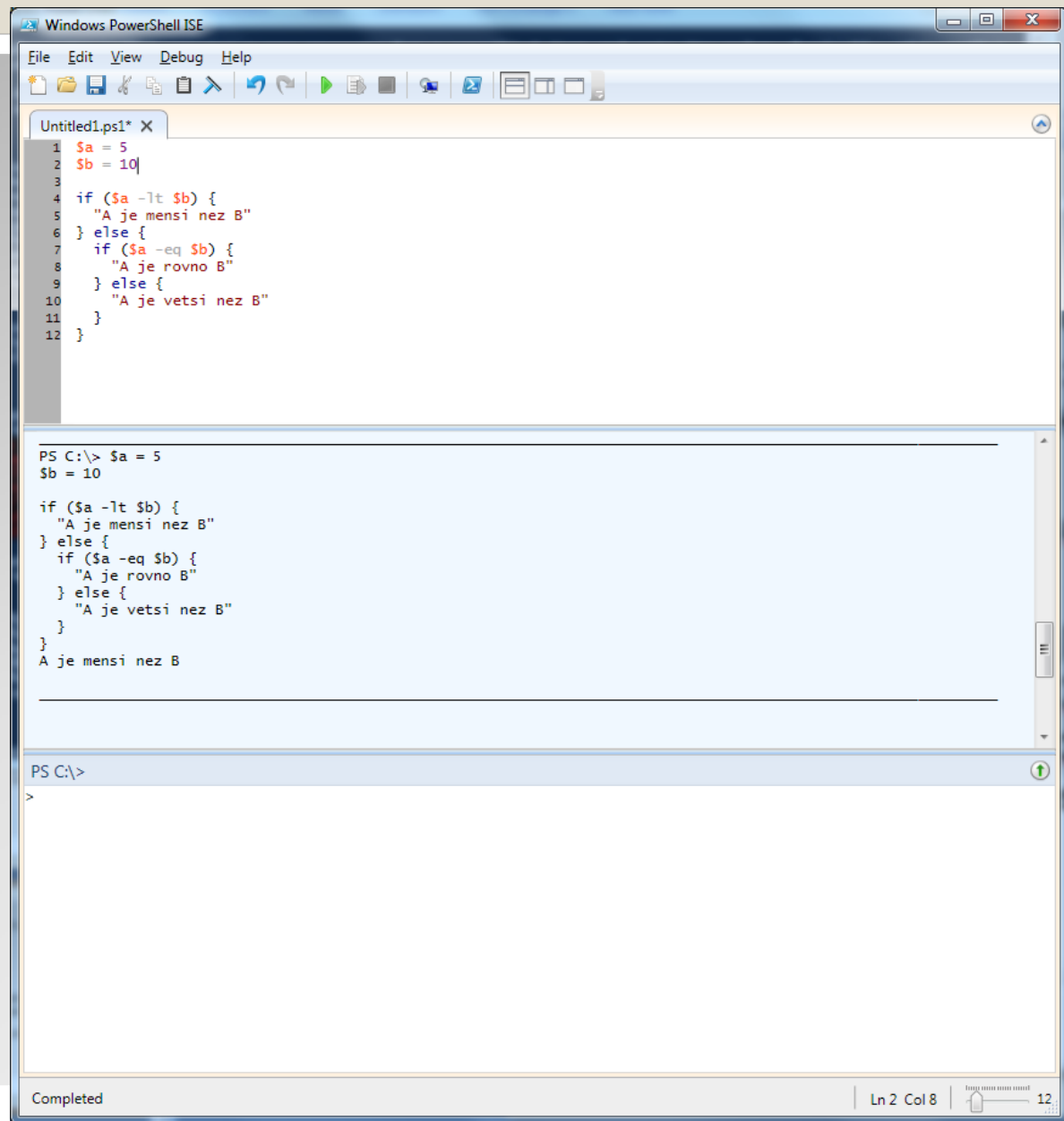
Příklady

- `Mkdir C:\Tmp`
 - Vytvoří složku „Tmp“ na oddíle C:
- `Set-Location C:\Tmp`
 - Přejde do složky „C:\Tmp“
- `CD C:\Tmp`
 - Totéž, s využitím aliasu „CD“
- `Get-ChildItem C:\Windows -Recurse -Force | Sort-Object length -Descending | Select-Object -first 1`
 - Vypíše největší soubor ze složky Windows

Příklady

- Set-Location hklm:
 - Nastaví registr počítače (HKLM) jako providera
- Get-Childitem hklm:
 - Vypíše klíče registru HKLM
- LS hklm:
 - Totéž s využitím aliasu „LS“
- New-ItemProperty
HKLM:\SOFTWARE\Microsoft\Windows\CurrentVersion\Run -name "Kalkulacka" -PropertyType "String" -value "calc.exe,"
 - Vytvoří v uvedené cestě novou hodnotu typu String

Příklady



The screenshot displays the Windows PowerShell ISE interface. The top pane shows a script named 'Untitled1.ps1' with the following code:

```
1 $a = 5
2 $b = 10
3
4 if ($a -lt $b) {
5     "A je mensi nez B"
6 } else {
7     if ($a -eq $b) {
8         "A je rovno B"
9     } else {
10        "A je vetsi nez B"
11    }
12 }
```

The middle pane shows the execution output:

```
PS C:\> $a = 5
$b = 10

if ($a -lt $b) {
    "A je mensi nez B"
} else {
    if ($a -eq $b) {
        "A je rovno B"
    } else {
        "A je vetsi nez B"
    }
}

A je mensi nez B
```

The bottom pane shows the prompt 'PS C:\>' and a cursor, indicating the script has been executed. The status bar at the bottom indicates 'Completed' and 'Ln 2 Col 8'.

PowerShell ISE

```
$a = 5
```

```
$b = 10
```

```
if ($a -lt $b) {  
    "A je mensi nez B"  
} else {  
    if ($a -eq $b) {  
        "A je rovno B"  
    } else {  
        "A je vetsi nez B"  
    }  
}
```

Příklad základního skriptu

```
PS D:\pracovni\sws\2016-2017> .\skript.ps1
.\skript.ps1 : File D:\pracovni\sws\2016-2017\skript.ps1 cannot be loaded because running scripts is disabled on this s
ystem. For more information, see about_Execution_Policies at http://go.microsoft.com/fwlink/?LinkID=135170.
At line:1 char:1
+ .\skript.ps1
+ ~~~~~
+ CategoryInfo          : SecurityError: (:) [], PSSecurityException
+ FullyQualifiedErrorId : UnauthorizedAccess
PS D:\pracovni\sws\2016-2017>
```

- **Set-ExecutionPolicy**
 - Restricted, AllSigned, RemoteSigned, Unrestricted

Spouštění skriptu (about_signing)

- <http://technet.microsoft.com/en-us/library/ee221100.aspx>
- <http://technet.microsoft.com/library/dn425048.aspx>
- <http://blogs.technet.com/b/technetczsk/archive/2009/07/22/serial-windows-powershell-uvod-cast-1.aspx>
- <http://pechanec.cz/Default.aspx?pageid=7>

Zdroje se syntaxí