PowerShell alapok

2011. Október 4.

Parancsértelmezők

- A DOS-os `command.com` és utódja a `cmd.exe` nem segítik a hatékony script írást (ellenpélda Unix/Linux világ, `bash`)
- A Microsoft a 2000-es évek elején egy újfajta megközelítésű parancssori menedzsment megoldás kidolgozásába kezdett, ez lett a Powerhell (kódnevén „Monad”)
  - A PowerShell első publikus bemutatója 2003 szeptemberében volt
  - A megoldás képességeiben sok szempontból túlmutat a – méltán – nagy hírű Linux shell-ek szolgáltatásain
  - Erős .Net „kapcsolat”
- Irodalom (ingyenesen elérhető): Soós Tibor és Szerényi László, Microsoft PowerShell 1.0 rendszergazdáknak - elmélet és gyakorlat

  http://www.microsoft.com/hun/dl.aspx?id=a5b21b7c-3d64-4144-a44c-27a495dbab2c

PowerShell parancsok

- Indítása `powershell` parancs kiadásával
- A PowerShell környezetében többféle parancs stílust is használhatunk:
  - Hagyományos „DOS” belső parancsok (pl. DIR)
  - Unix-os parancsok (pl. ls)
  - PowerShell „saját” parancsai, az ún. Cmdlet-ek (pl. Get-ChildItem)
  - Szabványos Windows futtatható programok
  - A DOS és a Unix parancsok valójában alias-ok a PowerShell saját parancsaira (Cmdlet)

Cmdlet

- A PowerShell saját parancsai
- Leírásuk minden esetben ige-főnév formájú
  - Pl. Get-ChildItem vagy Get-Process
  - Get-Help get-*, Get-Help Get-Process -example
- A parancs paramétereinek neve kötött
- A parancs kimenete nem „sima” szöveg, hanem objektum!
- A parancsok összefűzhetők (kompozit parancsok)

Parancsok összefűzése

- Ránézésre hasonló, mint a hagyományos parancssor: **`dir | find „ARIS”`**
- Szöveg helyett azonban objektumok „közlekednek”

```
> Get-ChildItem | where-object { $_.Length -ge 1000 }
```

- Ami akár tovább is láncolható

```
> Get-ChildItem | where-object { $_.Length -ge 1000 } |
Sort-Object -property Length
```

- Melyek az egyes objektumok tulajdonságai?

```
> Get-ChildItem | get-member
```

Legfontosabb parancselemek és argumentumok

Ige	Főnév
Add	ChildItem
Get	Item
New	Process
Remove	Object
Set	Help ☺

Argumentum
Verbose
Debug
WhatIf
Confirm
Property

Néhány fontos parancs

Kategória	Parancsok
Mappák és fájlok	Get-Childitem, Get-Item, New-Item, Remove-Item, Move-Item, Copy-Item, Rename-Item, Invoke-Item New-Item –type file alma Get-Item alma Rename-Iten alma körte Get-Item körte Remove-Item körte
Folyamatok	Get-Process, Stop-Process
Dátum és idő	Get-Date, pl: Get-Childitem | where-Object { $_.LastAccessTime –lt (Get-Date).AddDays(-10) }
Eseménynapló	Get-Eventlog, pl: Get-Eventlog System –Newest 10
Egyéb	Get-Member, Get-Help, Get-Alias

Get-alias két nézete

- Get-alias dir

```
PS C:\> get-alias dir

CommandType     Name                                               Definition
-----------     ----                                               ----------
Alias           dir                                                Get-ChildItem
```

- Get-alias –definition Get-Childitem

```
PS C:\> Get-alias -definition Get-Childitem

CommandType     Name                                               Definition
-----------     ----                                               ----------
Alias           dir                                                Get-ChildItem
Alias           gci                                                Get-ChildItem
Alias           ls                                                 Get-ChildItem
```

Változók

- A változók neve $ jellel kezdődik
- A változókat nem kell deklarálni
- A változó értéke a név megadásával lekérhető

```
>$most=Get-Date
>$most
```

- A változók objektumokat tárolnak

```
>$megint=Get-Date
>$elteres=$megint-$most
>$elteres
```

Szöveges változók (System.String)

- Karekterlánc típusú, értékadáskor a szöveget aposztrófok vagy idézőjelek között kell megadni

  >$t=“alma”

- Ez is objektum!
- A karakterlánc metódusai és tulajdonságai:

  >$t | Get-Member → … **(lista)**

  >$t.Length → **4**

  >$t.ToUpper() → **ALMA**

  >$t.Replace(“al”,”fel”) → **felma**

Fontosabb karakterlánc műveletek

Művelet	Eredmény
Replace(<mit>,<mivel>)	Szövegrész cseréje
Contains(<szöveg>)	Szövegrész tartalmazásának vizsgálata
StartsWith(<szöveg>)	Karakterlánc elejének vizsgálata
EndsWith(<szöveg>)	Karakterlánc végének vizsgálata
Substring(<start>,<end>)	Szövegrész kivágása
Trim, TrimEnd, TrimStart	„felesleges” karakterek levágása
Split	Szöveg darabolása

Gyűjtemények

- Létrehozása
  - Parancssorból: `$b = “alma”, “barack”, “citrom”`
  - Parancs kimenete: `$c = Get-Childitem`
    `$d = $t.split()`
- A length értéke ilyenkor az elemek száma

  `> $b.length` → `> 3`

Objektumok kezelése

- Bejárás: ForEach-Object

```
> Get-ChildItem | ForEach-Object { $sum+=$_.Length }
```

- Szűrés: Where-Object

```
> Get-ChildItem | Where-Object { $_.Length –ge 1000 }
```

- Rendezés: Sort-Object

```
> Get-ChildItem | Sort-Object -property Length
```

- Kiválasztás: Select-Object

```
> Get-ChildItem | Sort-Object -property Length | Select-
  Object –First 5
>  "a","c","b","a","b","d","e" | select-object -Unique
```

- Csoportosítás: Group-Object

```
> Get-ChildItem | Group-Object Extension
```

Eredményül kapott listák szűrése

- Könyvtárak listázása

```
Get-Childitem | Where-Object { $_.PsIsContainer }
```

- Leállított szolgáltatások listázása

```
Get-Service | Where-Object { $_.Status –eq
“Stopped” }
```

- Adott nevű folyamatok listázása

```
Get-Process | Where-Object { $_.Name –like
“*svchost*” }
```

Operátorok

Operátor	Magyarázat	Operátor	Magyarázat
-eq	Egyenlő	-like	Karakterlánc helyettesítő karakteres vizsgálata
-lt	Kisebb mint	-notlike	
-gt	Nagyobb mint	-contains	Egy gyűjtemény tagja-e az adott objektum
-le	Kisebb v. egyenlő	-notcontains	
ge	Nagyobb v. egyenlő	-replace	String.replace
-not	Logikai tagadás	-and	Logikai és
-match	Karakterlánc regexp alapú vizsgálata	-or	Logikai vagy
-notmatch		-band	Bitenkénti és
		-bor	Bitenkénti vagy
		-xor	Logikai kizáró vagy

```
3 -lt 5
6 -ge 7
“kutya” -like “ut”’
“kutya” -like “*ut*”
```

Reguláris kifejezések

karakter	Illeszkedés karakterre	\w	Alfanum. Karakter
. (pont)	Egyetlen tetszőleges karakter	\W	Nem alnum
[xyz]	A zárójelben szereplőkből legalább egy	\s	Üres karakter
[x-y]	Tartomány	\S	Nem üres
[^xyz]	xyz kivételével bármelyik kar.	\d	Szám
^abc	Sor eleje	\D	Nem szám
abc$	Sor vége	Kif{n}	Kif. Ismétlődés n alkalommal
x*	X karakterből 0…n darab	Kif{n,}	Ismétlődés legalább n-szer
x?	X karakterből 0…1 darab	Kif{n,m}	n és m közötti számú ism.
x+	X karakterből 1…n darab	\kar	Jelentés elnyomása (pl. \.)

Példák reguláris kifejezésekre

```
"alma" –match "ma" → True
"alma" –match "^ma" → False
"alma" –match "^al" → True
"alma" –match "ma$" → True
"alma" –match "^[a-d]" → True
"alma" –match "^[e-g]" → False
"1992-08-20" -match "^\d{4}-\d{2}-\d{2}$" → True
"alma.txt" -match "^\w{3,5}\.txt$" → True
"korte.txt" -match "^\w{3,5}\.txt$" → True
"barack.txt" -match "^\w{3,5}\.txt$" → False
```

Vezérlési szerkezetek

- If (<feltétel>) { <then ág> } else { <else ág> }

```
>$a=12
>if ($a -lt 20) {“Kicsi”} else {“Nagy”} → Kicsi
>$a=42
>if ($a -lt 20) {“Kicsi”} else {“Nagy”} → Nagy
```

- For(<kezdő>;<feltétel>;<lépés>) {<ciklusmag>}

```
>for($b=1; $b -lt 8; $b++ ) { $b } → 1..7
```

Példák

Fájlokkal kapcsolatos példák

- Fájl tartalmának beolvasása

```
$content = Get-Content c:\tmp\alma.txt
```

- Feltételnek megfelelő sorok kiíratása

```
$content | Where-Object { $_ -match "^.{3,5}$" }
```

- Vagy

```
Select-String –pattern "^.{3,5}$" c:\tmp\alma.txt
```

- Adott szöveget tartalmazó fájlok (rekurzív)

```
Get-ChildItem -Recurse -Filter *.txt |
foreach-object { if (Select-string –quiet
"[0-9]{3}-[0-9]{4}" $_.FullName) { $_.FullName }}
```

Fájlok szűrése

- Méret szerint

```
Get-ChildItem | where-object { $_.Length -gt 100KB }
```

- Utolsó hozzáférés szerint

```
Get-ChildItem | where-object { $_.LastAccessTime -lt
(Get-Date).AddDays(-10) }
```

```
get-childitem | where-object { $_.LastAccessTime –lt
(Get-Date).AddDays(-10) } | ForEach-Object { remove-item
-whatif -confirm $_.FullName }
```

- Csoportosítás kiterjesztés szerint

```
Get-ChildItem | Group-Object Extension
```

Fájlok helyfoglalása

- Írassuk ki a \Windows könyvtárban és alkönyvtáraiban található „exe” fájlok helyfoglalását Gbyte mértékegységben!

```
> get-childitem -Recurse -filter *.exe \windows |
  foreach-object { $size += $_.Length }
> $size/1GB
```

- 5 legnagyobb fájl nevének kiírása a \Windows\Help könyvtár alatt (rekurzívan)

```
>get-childitem -recurse  \windows\help\ |
 Sort-Object -Desc Length | Select-Object -First 5
```

Fájlok Q&A

- **Q: munkakönyvtár kiíratása**
- `A:(get-location).Path vagy $pwd`
- **Q: Könyvtár létrehozása**
- `A: create-item –type directory alma vagy md alma`
- **Q: Fájl vagy könyvtár létezésének ellenőrzése**
- `A: test-path alma`
- **Q: Fájl vagy könyvtár törlése**
- `A: remove-item alma`
- **Q: Fájl vagy könyvtár átnevezése**
- `A: rename-item alma barack`
- **Q: Fájl vagy könyvtár másolása, mozgatása**
- `A: copy-item barack citrom`
- `A: move-item citrom eper`
- **Q: Hozzáférési jogok lekérése**
- `A: get-acl citrom`

Két lista összehasonlítása

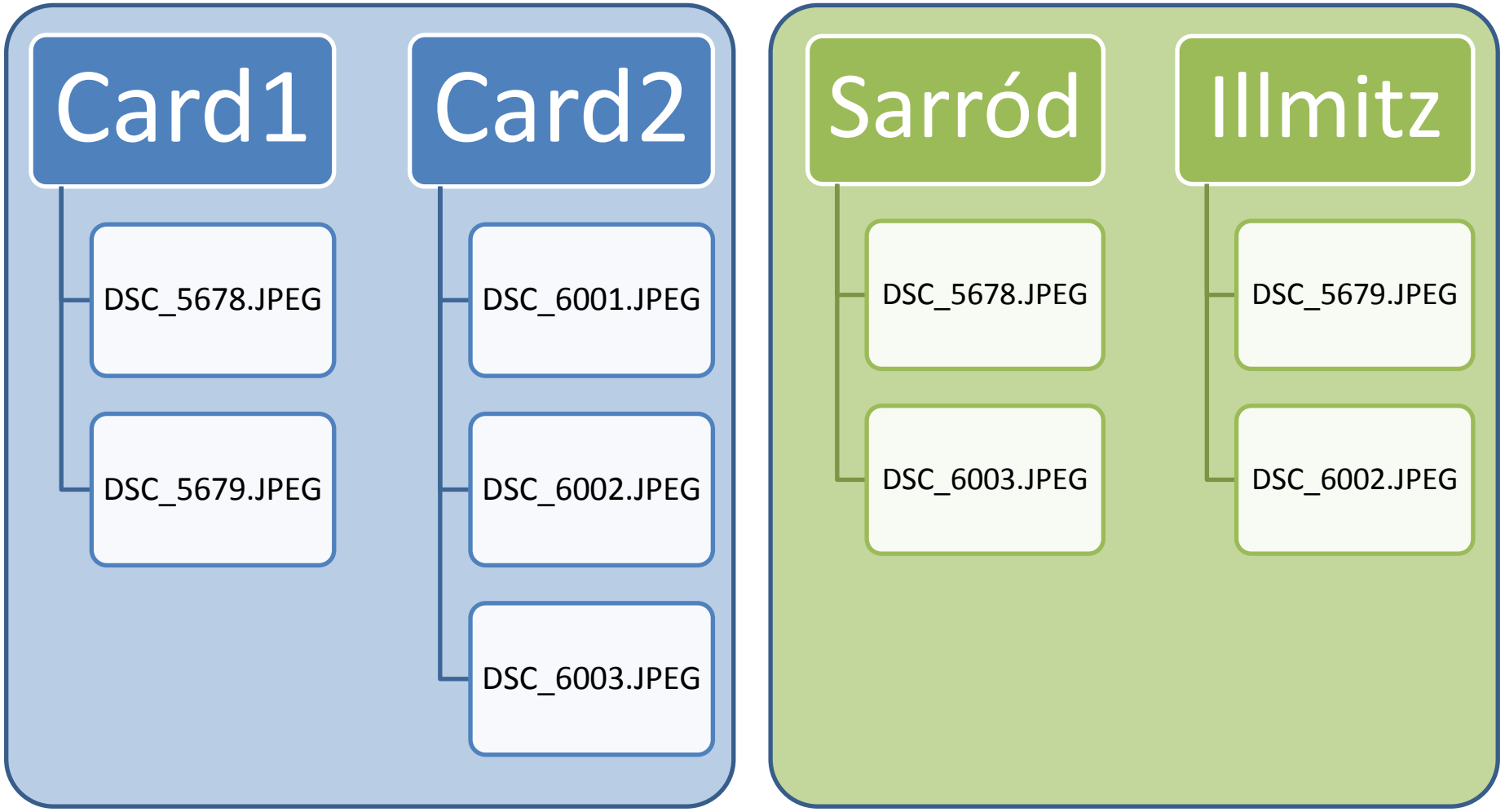
- Működés

```
$a = “alma”,”barack”,”dió”
$b = “alma”,”barack”,”citrom”,”dió”
Compare-Object $a $b
```

Feladat

- Adottak könyvtárak, bennük JPEG fájlok, mindegyik egyedi névvel (pl. DSC_1234.JPEG)
- Adott egy másik könyvtárstruktúra, többé-kevésbé azonos fájlokkal, de másfajta szervezésben
- Feladat: a két struktúrában található fájlok közötti különbségek megtalálása (csak név számít)

Feladat


Card1
DSC_5678.JPEG
DSC_5679.JPEG
Card2
DSC_6001.JPEG
DSC_6002.JPEG
DSC_6003.JPEG
Sarród
DSC_5678.JPEG
DSC_6003.JPEG
Illmitz
DSC_5679.JPEG
DSC_6002.JPEG

Megoldás

- Fájllista kinyerése

```
get-childitem -recurse | where-object { -not
  $_.PSIsContainer } | sort-object -property name
  | ForEach-Object { $_.Name }
```

- Összehasonlítás

```
$a = get-childitem –recurse c:\tmp\a …
$b = get-childitem –recurse c:\tmp\b …
Compare-object $a $b
```