

Thuis opdracht avond 3.

Het maken van een script. Gebruik hiervoor een script-editor. Hieronder de gratis editors

1. Powershell ISE `c:\windows\system32\WindowsPowerShell\v1.0\PowerShell_ISE.exe`
2. PowerGUI <http://www.powergui.org/downloads.jspa>
3. NotePad++ <http://notepad-plus-plus.org/download/v6.2.2.html>

Bij het starten van een powershell script (<filename>.PS1) moet je executionpolicy goed staan.

Met het commando get-executionpolicy kun je deze opvragen.

```
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS C:\Windows\system32> Get-ExecutionPolicy
Restricted
```

Er zijn meerdere manieren om het mogelijk te maken om Powershell scripts te draaien.

- Met het Set-ExecutionPolicy cmdlet kun je dit aanpassen.
Voor iedereen: `Set-ExecutionPolicy remotesigned`
Voor alleen jezelf: `Set-ExecutionPolicy -Scope CurrentUser remotesigned`
- Door het Powershell process op te starten met een tijdelijk execution beleid:
`Powershell -executionpolicy remotesigned`

```
PS C:\Windows\system32> Set-ExecutionPolicy remotesigned

Execution Policy Change
The execution policy helps protect you from scripts that you do not trust. Changing the execution policy might expose
you to the security risks described in the about_Execution_Policies help topic. Do you want to change the execution
policy?
[Y] Yes [N] No [S] Suspend [?] Help (default is "Y"): y
```

In de afgelopen avond hebben we een script gemaakt voor het uitlezen van lokale disks.

In dit script hebben we gebruik gemaakt van variabelen, het toevoegen van Comment Syntax (get-help), parameters, verplichte parameters (mandatory), een alias, parameter validatie en verbose output.

Opdracht

(Dit script draaien op een windows 7 SP1 machine)

```
Get-WMIObject Win32_OperatingSystem -ComputerName localhost -Filter "Buildnumber = 7601" |
Format-List -Property @{label='Operating System';expression={$_.Caption}},
OSArchitecture, BuildNumber, Version, MUILanguages, SerialNumber, SystemDrive, WindowsDirectory,
@{label='ComputerName';expression={$_.PSComputerName}},
@{label='Free Physical Memory (MB)';expression={$_.FreePhysicalMemory /1kb -as [int]}},
@{label='Free Space In Paging Files (MB)';expression={$_.FreeSpaceInPagingFiles /1kb -as [int]}},
@{label='Free Virtual Memory (MB)';expression={$_.FreeVirtualMemory /1kb -as [int]}},
@{label='Total Virtual Memory (MB)';expression={$_.TotalVirtualMemorySize /1kb -as [int]}},
@{label='Total Memory (MB)';expression={$_.TotalVisibleMemorySize /1kb -as [int]}},
@{label='Installation Date';expression={$_.ConvertToDateTime($_.InstallDate)}},
@{label='Last Boot Time';expression={$_.ConvertToDateTime($_.LastBootUpTime)}},
@{label='Current Local Time';expression={$_.ConvertToDateTime($_.LocalDateTime)}}
```

Uitleg bovenstaand script

Dit script maakt verbinding d.m.v. WMI met de class win32_operatingsystem Daarna filteren we op het buildnumber en maken we een geformateerde lijst met een selectie properties.

Voor sommige van deze properties hebben we wat aanpassingen gemaakt om de output vriendelijker te maken.

B.v. Free Physical Memory. De waarde die in WMI staat bij FreePhysicalMemory is 3581628.
D.m.v. de translatie Hash Array kan je een eigen naam meegeven en kan je een aangepaste waarde teruggeven d.m.v. een berekening.

```
@{label='Free Physical Memory (MB)';expression={$_.FreePhysicalMemory /1kb -as [int]}}
```

Of de lastbootime. De waarde die WMI geeft voor de LastBootUpTime is

20121121162039.125599+060.

Niet echt leesbaar. D.m.v. onderstaande hash converteren we deze output naar datetime notatie

```
@{label='Last Boot Time';expression={$_.ConvertToDateTime($_.LastBootUpTime)}}
```

Open je script editor en neem bovenstaand script over.

Save dit script bijvoorbeeld naar c:\demo\get-OSInventory.ps1

(het script is als get-OSInventory.txt ook gemaild)

Opdracht 1.

- Parametiseer het script met de volgende parameters
 - \$computername en maak de default waarde de localhost
 - \$buildnumber met als default waarde 7601 (7601 is Windows 7 SP1)

Opdracht 2

- Start het script vanuit powershell
- Werkt het script?
- Start het script nogmaals maar nu met de parameters -computername en -buildnumber
- Werkt het script?
- Krijg je nu output? Als het goed is niet.

Opdracht 3

- Voeg aan het script comment syntax toe.
- Vraag de help op van je script: get-help Get-OSInventory.ps1 -full
- Krijg je de help files te zien?
- Voeg een extra voorbeeld toe aan de comment syntax

Opdracht 4

- Maak de \$computername parameter verplicht (Mandatory)
- start het script zonder parameters. Krijg je de vraag om een parameter value in te vullen?
- voeg een helpmessage toe aan deze parameter.

Opdracht 5

- Maak een alias OSnumber aan voor de parameter buildnumber
- run het script met de parameter alias.
- Maak een alias hostname aan voor de parameter computername.
- run het script met de parameter alias.

Opdracht 6

- Maak een validatieset voor de parameter Buildnumber. Zorg dat het script alleen de buildnumbers 2600, 3790, 7600, 7601 en 9200 accepteert.
 - 2600 = Windows XP Professional
 - 3790 = W2K3 Enterprise
 - 7600 = W2K8 R2
 - 7601 = Windows 7 Sp1
 - 9200 = Windows 8
- Run het script met de buildnumber parameter 1234
- Krijg je een foutmelding?

- Run het script met de buildnumber parameter 2600 of 3790 of 7600.
- Krijg je een foutmelding?

Opdracht 7

- Voeg een aantal verbose meldingen toe
- Run het script met de parameter verbose.
- Krijg je de verbose meldingen te zien

Opdracht 8

Rename het script naar .txt en mail het naar ons.