Powershell
5. Scripts

Computersystemen 2 - Windows

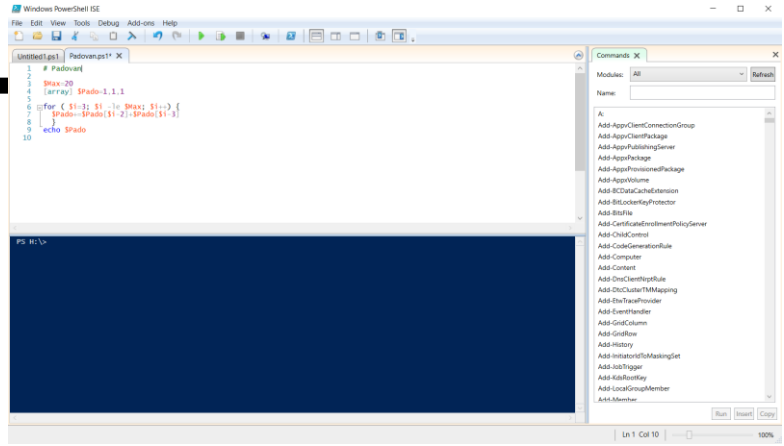
KdG Karel de Grote Hogeschool

13/03/2025

Overzicht

- Scripts
- Controle structuren
- Input en Output
- Script parameters
- Functies
- Oefeningen

Scripts



Tekstbestand, extensie .ps1

PowerShell ISE

Veiligheid: Niet dubbelklikken, volledig pad

- PS C:\> C:\test.ps1
- PS C:\> C:\test
- PS C:\> .\test.ps1
- PS C:\> .\test
- powershell C:\test.ps1

Restricted, AllSigned, RemoteSigned, Unrestricted

- PS C:\> set-executionpolicy unrestricted

Overzicht

- Scripts
- Controle structuren
- Input en Output
- Script parameters
- Functies
- Oefeningen

Controle structuren

if (*conditie*) {...} **elseif** (...) {...} **else** {...}

switch (*$var*) { *value* {...} *value* {...} **default** {...} }

while (*conditie*) {...}

do {...} **while** (*conditie*)

do {...} **until** (*conditie*)

foreach (*$var* **in** *$collection*) {...}

for (...; ...; ...) {...}

exit

Overzicht

- Scripts
- Controle structuren
- Input en Output
- Script parameters
- Functies
- Oefeningen

Input / Output

Van/naar terminal

- Write-Host "--------------"
- $a = Read-Host "Geef input"
- Clear-Host

Naar window

- Get-Process | Out-GridView

Naar window en selecteren

- Get-process | Out-GridView -OutputMode Single
- $a = Get-Process | Out-GridView -OutputMode Multiple
- $a

Overzicht

- Scripts
- Controle structuren
- Input en Output
- Script parameters
- Functies
- Oefeningen

Script parameters - array ("the Linux way")

$args: array van parameters

```
# Maak-Deling
# eerste parameter: deeltal
# tweede parameter: deler
if ($args.count -ne 2)
  {Write-Host "GEBRUIK: Maak-Deling deeltal deler"}
else
  {Write-Host ($args[0] / $args[1])}
```

Parameters hebben geen naam

```
.\Maak-Deling.ps1 7 2
```

Script parameters - parameter blok (beter!)

Parameter blok: mogelijkheid om default waarden te geven

```
# Maak-Deling2.ps1
# Default waarde van deeltal en deler is 1
Param(
  [int]$deeltal=1,
  [int]$deler=1
  )
Write-Host ($deeltal / $deler)
```

Parameters hebben een naam

```
.\Maak-Deling2.ps1 -deler 2 –deeltal 7
.\Maak-Deling2.ps1  2
```

Overzicht

- Scripts
- Controle structuren
- Input en Output
- Script parameters
- Functies
- Oefeningen

Functies

```
function FunctionName ($par1, $par2) {
    code
    return $waarde
    }
```

FunctionName value1 value2 -> Hier geen haakjes !!!

FunctionName –par1 value1 –par2 value2

Probeer:

- Schrijf functie FahrenhToCelsius met ($fahrenh – 32) / 1.8
- FahrenhToCelsius 40

Functies met parameter blok

mogelijkheid om default waarden te geven

```
function FunctionName   {
    Param(
        [string] $par1="default",
        [switch] $par2=$false
        )
    code
    return $waarde
    }
```

Overzicht

- Scripts
- Controle structuren
- Input en Output
- Script parameters
- Functies
- Oefeningen

Oefening 5.1: Get-FileContent.ps1

Get-Content filelist.txt

```
c:\windows\win.ini
c:\windows\system.ini
```

Inhoud van alle files in filelist.txt

- Get-Content filelist.txt | Foreach-Object { Get-Content $_ }
- cat filelist.txt | % { cat $_ }

Maak een script dat hetzelfde doet.

Stel `$filelist = Get-Content filelist.txt`

Maak 2 versies door gebruikt te maken van:

1. foreach (... in ...) {...}
2. for (...; ...; ...) {...}

Oefening 5.2: Maak-Klas.ps1

Get-Content klassen.csv # maak deze aan

```
C:\temp\klas1,student,8
C:\temp\klas2,leerling,5
```

Maak een script dat voor elke lijn in deze file:

- Folder aanmaakt met naam zoals in 1ste veld (bv. `klas1`) *
- In deze folder de lege files `student1` t.e.m. `student8` aanmaakt (idem voor andere lijnen)
- Maak hiervoor gebruik van een functie “Klas”, met 1 parameter, een klas-object. Voor elke lijn in de csv-file wordt de functie “Klas” aangeroepen.
- Als de optie –`Remove` meegegeven wordt, worden de folders uit de csv file verwijderd

* Mocht je geen c:\temp folder hebben, maak deze eerst manueel aan

Oefening 5.3

Maak een file menu.csv aan zodat Get-Content menu.csv

```
1,Calc
2,Notepad
3,MSpaint
4,Stoppen
```

Probeer:

- $menu= Import-Csv -header "Nbr","Name" menu.csv
- $menu
- $menu[1]
- $menu[1].Name

Oefening 5.3: Toon-Menu.ps1

Schrijf een script genaamd Toon-Menu.ps1 dat in een lus:

- de inhoud van menu.csv (zie vorige slides) op het (blanco gemaakt) scherm toont *
- vraagt om een nummer in te lezen (nummers zijn altijd 1, 2, 3,…)
- het corresponderende programma uitvoert
- uit de lus springt bij het nummer dat correspondeert met “Stoppen”

Tip: gebruik de call operator (zie hoofdstuk 1)

*: indien de output niet op het scherm geflusht wordt, kan je pipen naar Format-Table