# -----------------------------------

# Basic Commands

# -----------------------------------

#start powershell from explorer

Get-help Get-Item

Get-help Get-Item -ShowWindow

Get-Command -Verb update

Get-Item .\

Get-ChildItem .\

Get-ChildItem .\ -Recurse

Get-Content .\demo-basic-commands\folder2\hello.txt

# -----------------------------------

# Pipe lines Commands

# -----------------------------------

# Not implemented ?

# Get-ChildItem -Path C:\WINDOWS\System32 | Out-Host -Paging

# Show paths

Get-ChildItem .\ -Recurse | Out-GridView

# Print selected entries

Get-ChildItem .\ -Recurse | Out-GridView -PassThru | foreach { Write-Host $\_.Name -ForegroundColor Green }

# -----------------------------------

# Aliases

# -----------------------------------

function Write-SayHello() {

Write-Host "Hello, how are you" -ForegroundColor Yellow

}

Set-Alias hello Write-SayHello

Get-Alias

Get-Alias he\*

# -----------------------------------

# Variables

# -----------------------------------

$myVar

$myVar = Get-ChildItem .\ -Recurse

$env:APPDATA

# -----------------------------------

# Comparison Operators

# -----------------------------------

8 -eq 7

5,6,7,8 -lt 7

"Euricom Cruise!" -match "Euri" # Match by regex => true

"Euricom Cruise!" -contains "Euri" # Exact match => false

5 -in 5,4,3,2,1

"It's Sunny on the cruise" -replace "S", "F" # case in-sensitive by default

"It's Sunny on the cruise" -creplace "S", "F" # case sensitive

# -----------------------------------

# Loops

# -----------------------------------

# For loop

for ($i = 1; $i -lt 5; $i += 1)

{

Write-Host $i

}

# Do While loop

$i=1

Do

{

$i

$i++

} While ($i -le 10)

# Foreach loops

Get-ChildItem .\ -Recurse | foreach { Write-Host $\_.Name -ForegroundColor Green }

Get-ChildItem .\ -Recurse | %{ Write-Host $\_.Name -ForegroundColor Green }

foreach ($item in Get-ChildItem .\ -Recurse)

{

Write-Host $item

}

# -----------------------------------

# Functions

# -----------------------------------

function HelloWorld() {

Write-host "Hello world"

}

# Functions need to start with verbs

Get-Verb

# New name

function Write-HelloWorld() {

Write-host "Hello world"

}

# Name parameter

function Write-HelloWorld($name) {

Write-host "Hello $name"

}

Write-HelloWorld -name 'Thomas'

Write-HelloWorld 'Thomas'

# place parameter

function Write-HelloWorld($name, $place) {

Write-host "Hello $name from $place"

}

Write-HelloWorld 'Thomas' 'Wachtebeke'

# Function return variables

# the special one

function Calc-Something($a,$b) {

$a

$b

return $a + $b

}

# Ensur only the sum is returned

function Calc-Something($a,$b) {

$a | Out-Null

$null = $b

return $a + $b

}

# Args variable

function Calc-Something($a,$b) {

$args

return $a + $b

}

# -----------------------------------

# Modules

# -----------------------------------

# Basic module

Import-Module -Name ".\BasicScriptModule.psm1" -Force

# Verify loaded module

Get-Module

# Advanced module

cd C:\git\Tools.PowerShell\PSModules\PowerShell.Tools.SQL

Import-Module .\PowerShell.Tools.SQL.psm1

Get-Module

# Get commands available in a module

get-command -Module 'PowerShell.Tools.\*'

cd C:\git\cruise-basic-powershell

# Module path

$env:PSModulePath -split ';' #users path

[Environment]::GetEnvironmentVariable("PSModulePath", "Machine") -split ';' #machine

# List available modules

Get-Module -ListAvailable

# -----------------------------------

# $Profile

# -----------------------------------

$profile

$profile.CurrentUserCurrentHost

Test-Path $profile

$PROFILE | Format-List \* -force #Show all paths to the profiles

psedit $profile

psedit $profile.CurrentUserAllHosts

# -----------------------------------

# $Powershell Core

# -----------------------------------

# Pipelines like the always work

Get-Process | Sort-Object CPU -Descending | Select-Object -First 5

# Windows alias on Mac

dir

# .NET functions

[math]::pi

(Get-Date).AddDays(7)

# Invoke rest requests

(Invoke-WebRequest -Uri 'http://jsonplaceholder.typicode.com/posts?userid=1' -Method Get).Content | ConvertFrom-Json

# Multiple profiles are also supported

$PROFILE | Format-List \* -force #Show all paths to the profiles