Scripting under Windows

Wolfgang Aigner
NTx BackOffice Consulting Group GesmbH
www.ntx.at

Agenda

- Scripting History
- Windows Scripting Host
  - Samples and ideas about WSH
  - Limits of WSH
  - How to learn - MOC2433 3 days training
- Windows Management Instrumentation Interface
  - Benefits and samples
  - Limits
  - How to learn - MOC2439 2 days training
- Powershell
  - Resouces, Environment
  - Samples
  - How to Learn - PS Course 2 Days

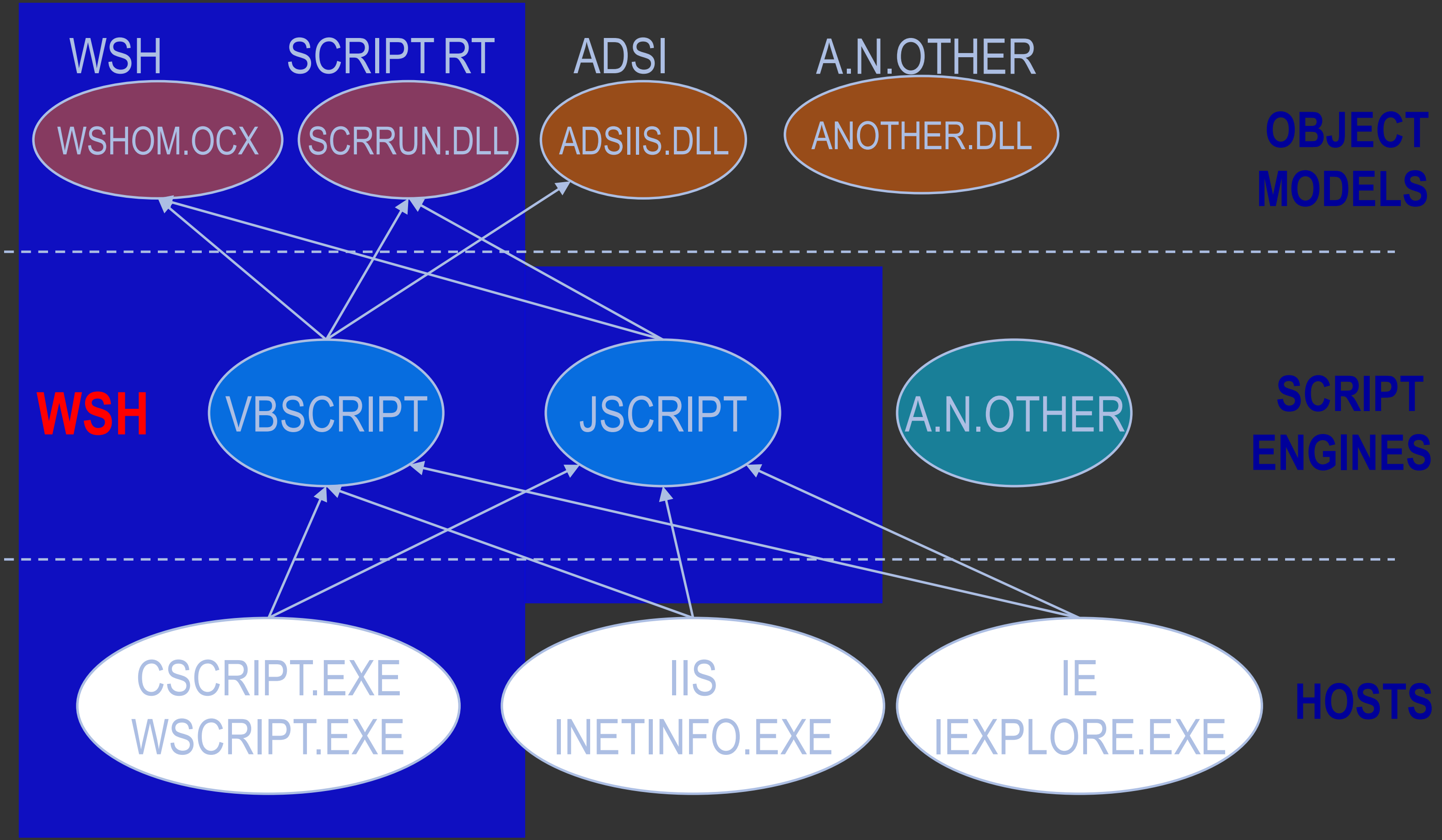
History

- Windows95/97 Command.exe *.bat
- Windows2000/XP cmd.exe *.cmd
- Windows2000/XP Windows Scripting Host (WSH)
- Windows Management Instrumentation Interface

WSH Windows Scripting Host

- On every system by default
- Easy to use
- Basic as an language
- Use all of the com objects
- vbs as a normal file
- Vbe as encrypted file
- // Parameter for the scripting host
- wscript, cscript

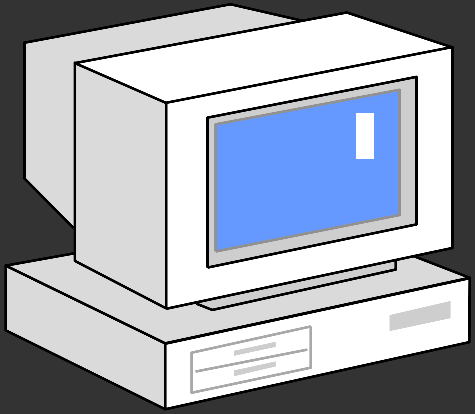
The WSH Environment



Windows Scripting Host

- Course Material MOC2433
- Object Model in WSH Module 2
- Active Directory Service Interface Module 5
- Simple Solution - scriptomatic
- Resourcen
- http://www.microsoft.com/technet/scriptcenter/tools/admatic.mspx

Windows Management Instrumentation

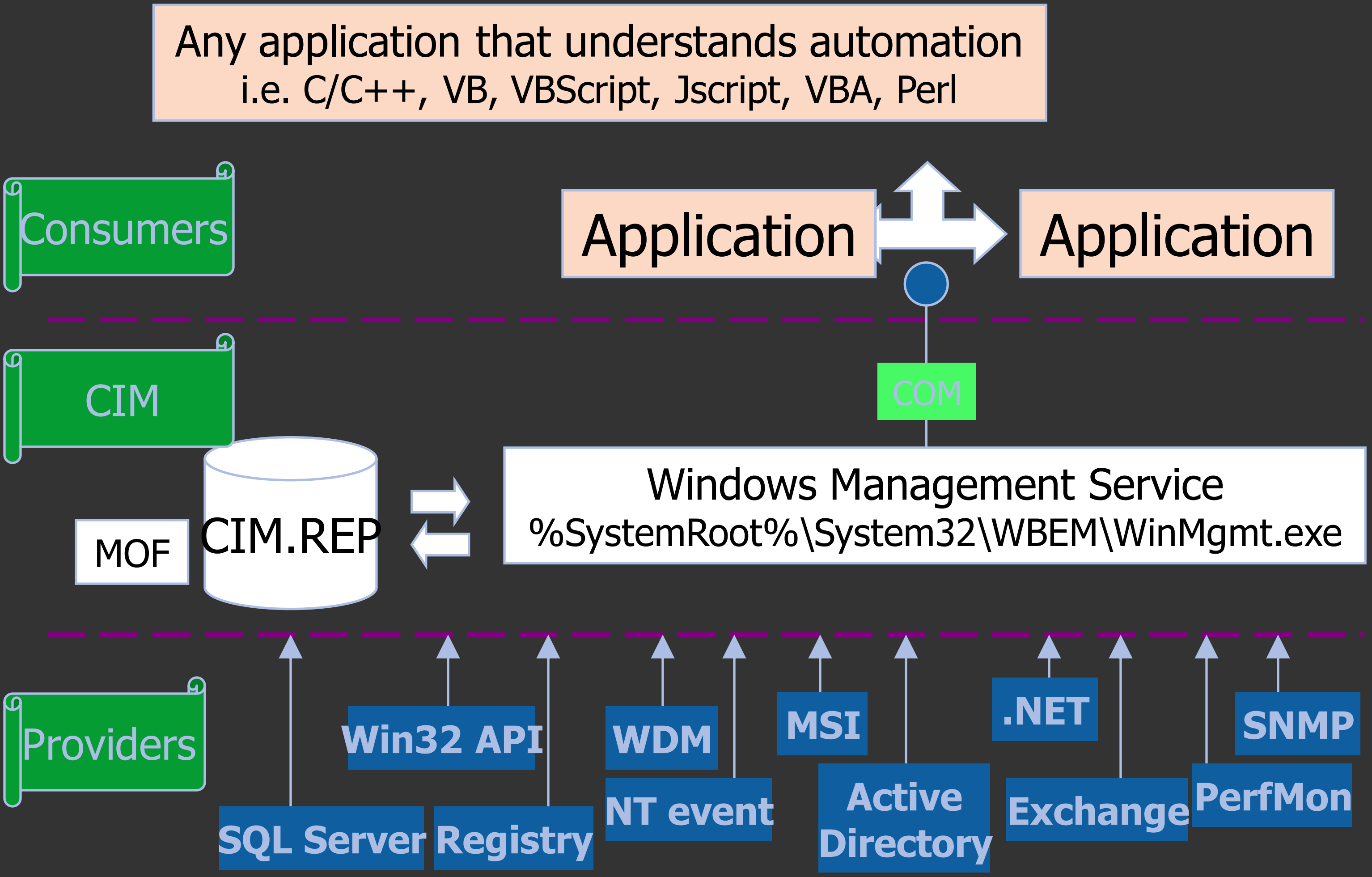


Windows Management Instrumentation Interface
Or
Manage your System easyly

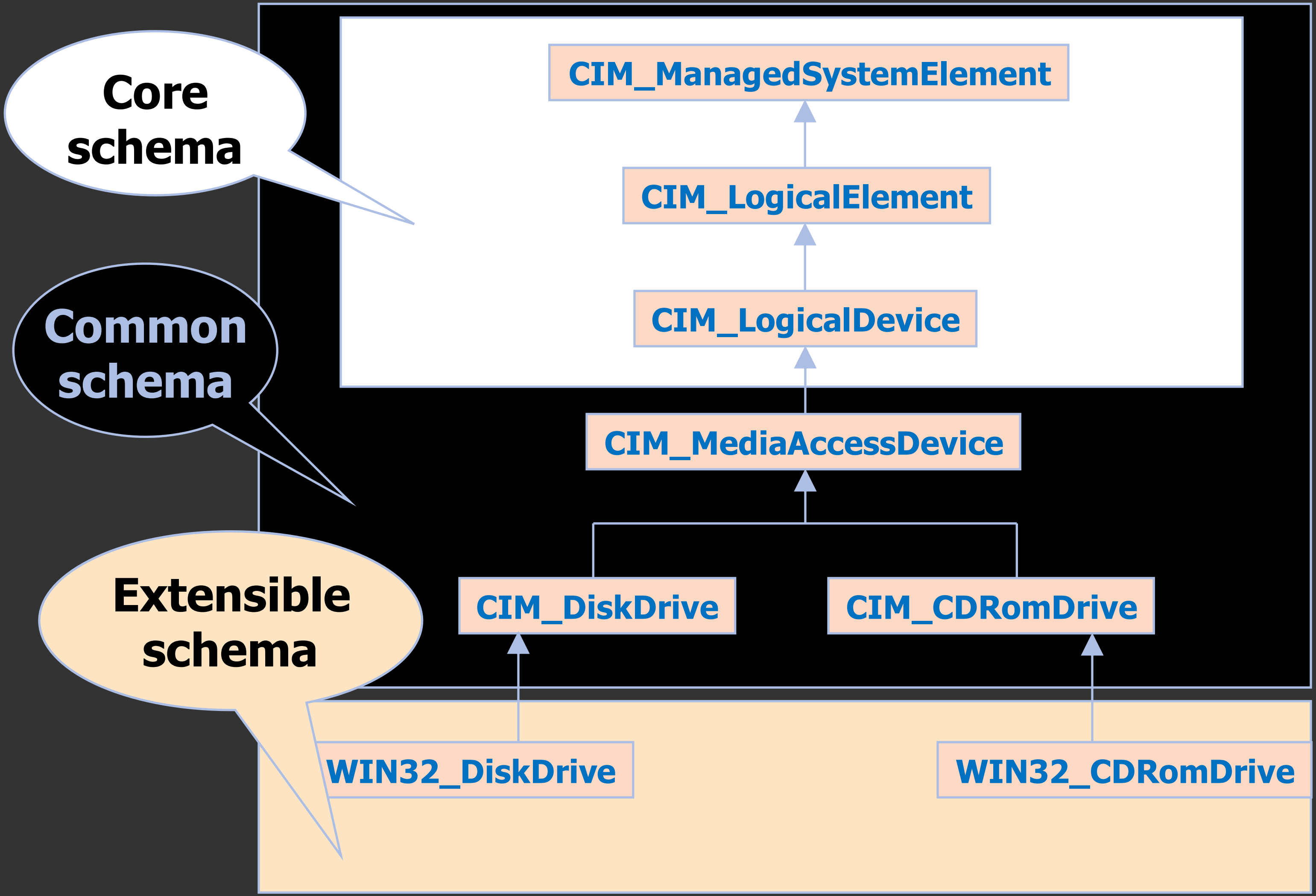
Windows Management Instrumentaton

- Database of all Computer Prop
- Very Detailed
- Similar Query Language than SQL - WSQL
- Retrieve Object
- Can also react on System Events
- Easy to use in WSH

WMI Architecture


Any application that understands automation
i.e. C/C++, VB, VBScript, Jscript, VBA, Perl
Consumers
Application
Application
CIM
COM
MOF
CIM.REP
Windows Management Service
%SystemRoot%\System32\WBEM\WinMgmt.exe
Providers
Win32 API
WDM
MSI
.NET
SNMP
SQL Server
Registry
NT event
Active Directory
Exchange
PerfMon

CIM Schemas


Core schema
Common schema
Extensible schema
CIM_ManagedSystemElement
CIM_LogicalElement
CIM_LogicalDevice
CIM_MediaAccessDevice
CIM_DiskDrive
CIM_CDRomDrive
WIN32_DiskDrive
WIN32_CDRomDrive

WMI

- Course Material MOC2439
- WMI Query Module 3

WMI

- WMI Tools
- CIM Studio
- WMI CodeGenerator
- WMI Resourcen http://msdn2.microsoft.com/en-us/library/aa286547.aspx

Powershell

How to make Life of an Admin easy !

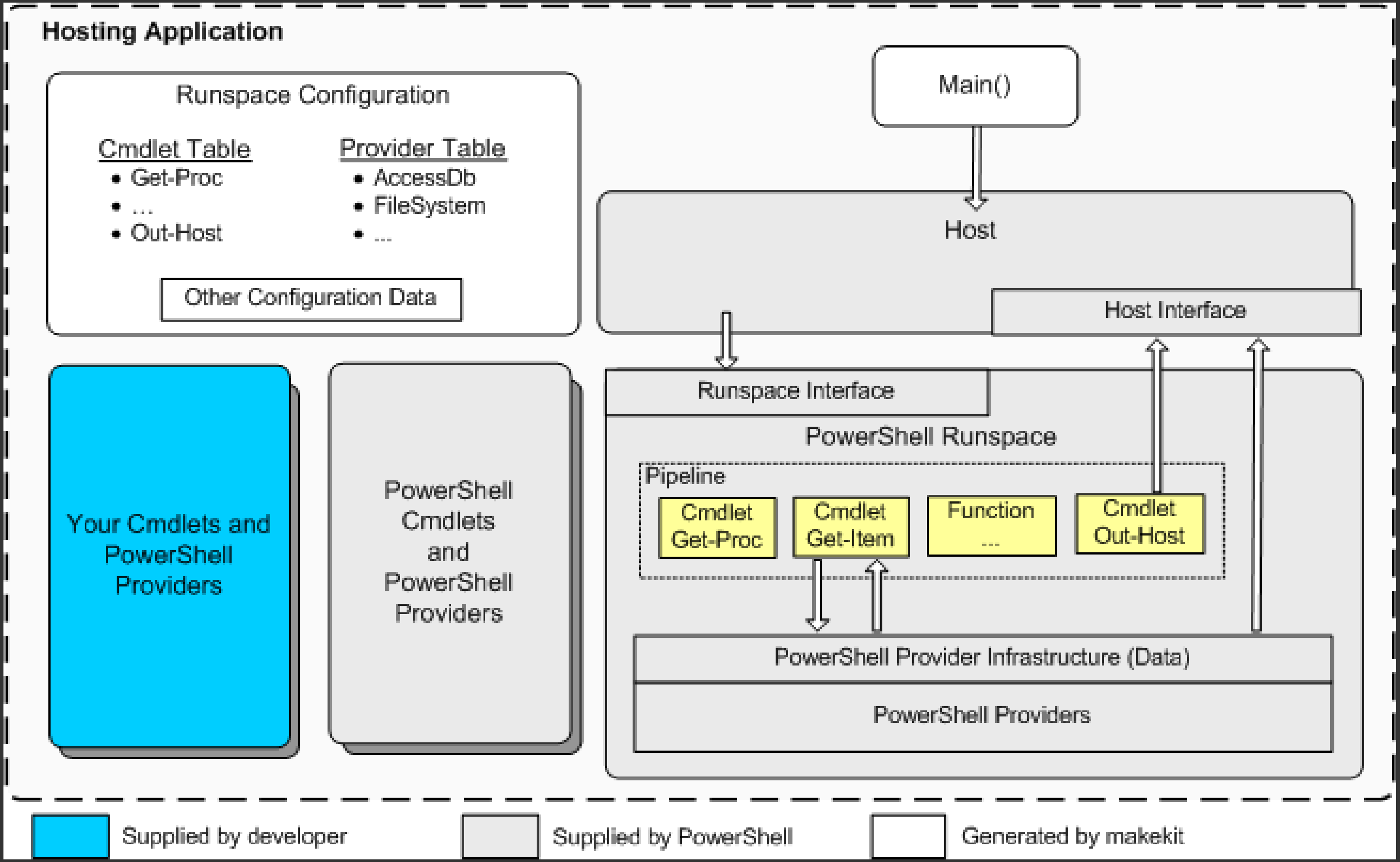
Powershell

- Hast to be installed on Windows 2003 an XP
- Part of Windows Server 2008 and Vista
- total new language
- new idea behind the language
- everything is a object
- objects have all properties and methods

Resources PS

- **Handouts from the Presentation**
- **PowerShell Language Quick Reference**
- **PowerShell User Guide.doc**
- Resourcen
- http://msdn2.microsoft.com/en-us/library/ms950396.aspx

PowerShell Environment



Help

- Get-help cmdlet
- Cmdlet -?
- Help cmdlet

Basics

- Start off the Shell powershell.exe
  - Parameter Powershell -h
- Execution Policy
  - In Registry
  - HKLM:\software\microsoft\PowerShell\1\ShellIds\Microsoft.PowerShell
  - AllSigned - Scripts must be signed
  - Unrestricted – must not be signed
  - Get-ExecutionPolicy
  - Set-ExecutionPolicy

Basics

- Command-lets
  - small pieces of a program
  - Start with a verb
  - Subject
  - Sample: Get-date
  - - for the parameter
  - Get-command lists all command-lets

Basics

- **Usefull commandlets**
  - **Get-Help**
  - **Get-Command**
    - Get-command get*
    - Get-command –verb get
    - get-command –type cmslet
      - Function
      - Filter
      - Cmdlet
      - ExternalScript
      - Application
      - Script
      - All

Basics

- Usefull commandlets
  - Get-member
    - Lists all members of an object
  - New-object
    - Creates an object of the specified type
    - New-object system.date

Basics

- All existing command line prg can be executed
  - e.g. $rst=netstat
  - Write-output $rst
  - Current path not in the path of the powershell
  - Use .\ for current path

Basics

- **Execution Order**
  - The PowerShell attempts to resolve commands in the following order:
  - Aliases
  - Functions
  - Cmdlets
  - Executables
  - Scripts
  - Normal files

Basics

- Com Object integration
- still can use com objects
  - Cool example with Internetexplorer
    - PS> $ie = new-object -com internetexplorer.application
    - PS> $ie.navigate2("http://www.msn.com/")
    - PS> $ie.visible = $true

Basics

- Pipe
  - |
  - Easier and faster
  - Pipe Object - not only text
  - Properties
  - Methods
  - All is available

Variables

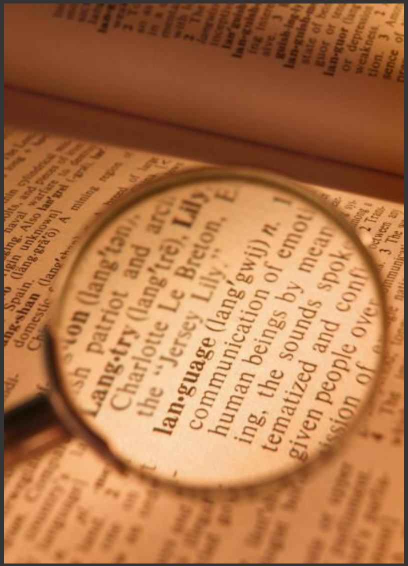
- Variables
  - All, object, strings, …
  - $
  - {} optional
  - Samples

```
PS> $stringVariable = "This is a string"
PS> $stringVariable
This is a string
PS> $!@#$%^&*() = "This is a non-traditional variable name"
Invalid variable reference. '$' was not followed by a valid variable name
    character. Consider using ${} to delimit the name.
At line:1 char:1
+ $ <<<< !@#$%^&*()
PS> ${!@#$%^&*()} = "This is a non-traditional variable name"
PS> ${!@#$%^&*()}
This is a non-traditional variable name

PS> ${stringVariable} = "This is a string"
PS> ${numberVariable} = 4
PS> ${processes} = Get-Process
```

Language

- Expression mode
  - 2+2
  - $test + 1
- Command mode
  - No quotes, everything is a string
  - Write-host 2+2
  - Write-host get-date
  - Write-host (get-date).day
- Boolean
  - False = 0,$null,""
  - True = every other string
  - $true, $false to be shure



Language

- Keywords are Context Sensitiv
  - foreach command
  - foreach alias for-eachobject

```
PS> foreach($i in 1,2,3,4,5) { $i }
1
2
3
4
5
PS> 1,2,3,4,5 | foreach { $_ }
1
2
3
4
5
```

Conditions

- If/else if/else

```
if (<expression1>)
        {<script_block1>}
    [elseif (<expression 2)
        {<script_block2>}]
    [else
        {<code_block3>}]
```

```
if ( test-path C:\Temp\MyDirectory )
{
    "C:\Temp\MyDirectory exists"
}
else
{
   new-item -type directory C:\Temp\MyDirectory
}
```

Looping

- **While**

```
while (<condition>){ <command_block> }
```

- **Do While**

```
$var = 0
do {
    $var++
    "var is $var"
} while ( $var -lt 2)
```

- **Do Until**

```
$var = 0
do {
    $var++
    "var is $var"
} until ( $var -gt 2)
```

-

Looping

- **Foreach**

```
foreach ($<item> in $<collection>){<command_block>}
foreach ($file in get-childitem)
{
    $file
}
```

- **For**

```
for (<init>; <condition>; <repeat>)
{<command_block>}
$i=1
for(;$i -le 10;$i++){$i}
```

Arrays

- **Working with arrays**

```
PS> $A = 22,5,10,8,12,9,80
PS> $B = 5..8
PS> $a.gettype() # gets type
PS> $a[0] # stat with 0
PS> $a[-3..-1] # last element to 3rd last
PS> $a # lists all elements
PS> foreach ($value in $a) {$value}
PS> $a += 200 # adds new element to array
PS> remove-item variable:a # deletes the array
```

Format

- Format-list
- Format-table
- Format-wide
- Format-custom

Alias

- Alias
  - Get-alias
  - Set-alias
    - Set-alias np notepad.exe

    - Function sw { set-location c:\work }
    - Set-alias pf sw

Provider

- Provider
  - Get-psprovider
  - get-psdrive
    - This command lists all the existing drives and the PowerShell Providers supporting those drives.
- Usage
  - Get-content env:username
  - $env:username
- Own drives
  - PS> new-psdrive -name etc -root c:\windows\system32\drivers\etc -psp filesystem

```
– Name      Provider         Root     CurrentLocation
– ----      --------         ----     ---------------
– Etc       Microsoft.... C:\windows\system32\drivers\etc
```

Extend the shell

- Load snapins
  - Add-pssnapin
- See whats loaded
  - Get-pssnapin

Courses for Scripting

- Windows Scripting Host
  - MOC2433
  - 3 days for Adminstrators
- Windows Management Instrumentation Interface
  - MOC2439
  - 2 days only WMI scripts based on vbscript
  - Useful after you have done MOC2433
- Powershell
  - Our own Material
  - 2 days course Powershell in Detail
  - Lots of samples

Thanks !

Any Questions ?

Buffet and Smalltalk !

NTx

BackOffice
Consulting Group

www.ntx.at
1010 Wien, Sterngasse 11
Tel +43 1 532 40 32