

Module 3

Modules Create Module



1

Modules



What is a module?

A module is a package that contains Windows PowerShell commands, such as cmdlets, providers, functions, workflows, variables, and aliases." (get-help about_modules)

- Enhancement of PowerShell
- Importable
 - Manually
 - Automatically
- Cmdlets
 - *-module

Get-Module
Get-Module -ListAvailable

Import-Module SmbShare [-force] [-PSSession \$session]
Import-Module SmbShare [-Prefix ADVPS] [-AsCustomObject]
Import-Module SmbShare [-Alias] [-Function] [-Variable]

Remove-Module

Page ■ 2

SERVICES

2

Create Custom Module



Custom modules are very often referred to 'tools'.

- 1 Choose a name for the module
- 2 Decide where to save the module and create a folder
- 3 Create the script
- 4 Create and edit the manifest
- 5 Import and test the module



Page • 3

3

1 - Choose a name for the Module



- The name is use for
 - Module folder
 - Module file (*.psm1)
 - Manifest file (*.psd1)
- Consists of
 - · Letter, numbers & underscore
 - Avoid spaces



Page ■ 4

4

2 - Module location



Decide where you want to save your module

\$Env:PSModulePath

- Form these paths modules are loaded automatically
- Add custom path(s)

```
$Env:PSModulePath
$Env:PSModulePath.Split(";")

C:\Users\Username\Documents\WindowsPowerShell\Modules
C:\Program Files\WindowsPowerShell\Modules
C:\Windows\system32\WindowsPowerShell\v1.0\Modules\
```

Other locations must be given

No automatically loading

Import-Module -Name c:\temp\custModule.psd1



Page ■ 5

5

3 - Create the Script



The Script is the core of a module

Modules provide

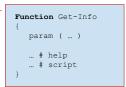
- Cmdlets, if the module is a compiled file (*.dll)
- Functions, if a function is written in PS-Code

Each module function is a script function

- Saved in *.psm1 or in referenced *.ps1 files
- More than one module function is possible, each is a function in PS-Code
- Other modules could be loaded
- Param, Begin, Process, End, ... everything is possible

Name of each function

- Use allowed verbs (get-verb)
- Otherwise warning
- Warning could be suppressed



Page • 6



4 - Create and Edit Manifestfile



A manifest allows you to configure some settings

- Load *.ps1xml automatically (format and type)
- Root module
- Information like author, company, version of module and PowerShell
- Load of prerequisites like module, scripts, assenblies
- Exportable functions, cmdlets, aliases and variables
- Created by cmdlet
- Edited by ISE e.g.

New-ModuleManifest -Path .\APS.psdl `
-Author 'DIST' `
-CompanyName 'DI Thomas Schleich' `
-ModuleVersion 1.0 `
-PowerShellVersion 3.0 `
-RootModule .\APS.psml



Page ■ 7

7

5 - Import and Test a Module



Get-Module

- -listavailable: shows all loadable modules
- -listavailable -refresh: reloads the list of all loadable modules

Cmdlet Import-Module

- -name (path + module-file): if you saved the *.psm1 in any location
- -force: unload the module and reload it again
- prefix: use a custom prefix for all nouns of your functions/cmdlets
 - . It overrules the prefix in the manifest.

Test

- Debug script before you create the module
- use: try ... catch ... finally, set-psdebug, integrated debugging, ...



Page • 8

Lab - Module



- Create a module with two functions
 - Check-DayTime
 - Hours 5-9: Good-Morning
 - Hours 19-22: Good-Evening
 - all others: Hello
 - Write-Hello
 - "\$(Check-DayTime \$Env:Username)
 - Create an alias hello for Write-Hello
- Only the function Write-Hello and the alias are public
- Module name is aPSGreetings
 - The module should be imported automatically.



Page • 9

9





Do You Have Any Questions?



Page ■ 10

10