PowerShell Modules

Martin Schvartzman

Principal Azure Consultant @ Microsoft

martin.schvartzman@microsoft.com

@martin77s

PowerShell Modules

What are modules, and how to use them?

Why should I create my own modules?

Module ingredients

Creating Modules

Release pipeline

What are Modules? How to use them?

“..A module is a package that contains PowerShell commands,

such as cmdlets, providers, functions, workflows, variables, and aliases.”

- Get-Module / Get-Module -ListAvailable
- $PSModulePath
- Import-Module / Remove-Module
- $PSModuleAutoLoadingPreference
- Find-Module / Save-Module / Install-Module
- Get-PSRepository (+Set) / Register-PSRepository (+Unregister)

Why should I create my own?

- Group related functions together
- Re-use “helper functions” (not exposed to the user)
- Share state between functions, but not with the user
- Improve discoverability:
  - Find-Module MyModule
  - Get-Command -Module MyModule
- Simplify distribution:
  - Install-Module MyModule
- Publish and get credit for your work

Module ingredients

- A Module Manifest
  - .psd1 file that describes your module
- A Root Module
  - .psm1 file. This is your PowerShell code that runs when

    the module is imported
- Exported (Public) Functions
  - These are the (advanced) functions an end user can run from our module

Module ingredients (cont’d)

- Private Functions
  - Optional “helper functions” to be used internally in our exported functions, that the end user shouldn’t see
- Formats
  - Optional format.ps1xml formats to help decorate your output
- .psm1-Help.xml file
  - Optional. For when using # .EXTERNALHELP <XML Help File> instead of using inline function comment-based help

Advanced Functions

- Use standard Verb-Noun for all public names
  - Get-Verb
- Prefix your Nouns to avoid conflicts
  - Company or product abbreviation
  - “PS” is reserved for PowerShell, none for the OS
- Parameters
  - Don’t prefix the parameters. It ruins tab completion experience
  - Property names == parameter names
  - Set- parameters should match object properties

Advanced Functions (cont’d)

- Enable Pipeline Binding

  $obj | Get-ObjectType

  Get-myTask –Id $id | Get-myJob | Get-myJobHistory

- Leverage Parameter Attributes

  Mandatory, ValueFromPipeline, Position, ParameterSetName

- ValueFromPipelineByPropertyName

  PSComputerName, VMName, __SERVER, MachineName

- Use the PowerShell streams

  Write-Output, Write-Warning, Write-Error, Write-Verbose

Creating Modules

- New-ModuleManifest

  Creates the hashtable manifest with the module’s metadata

  Export only the “public” functions

- Get-Command -Module

  Verify module properties and commands visibility

- Document your Module (about_Comment_Based_Help)

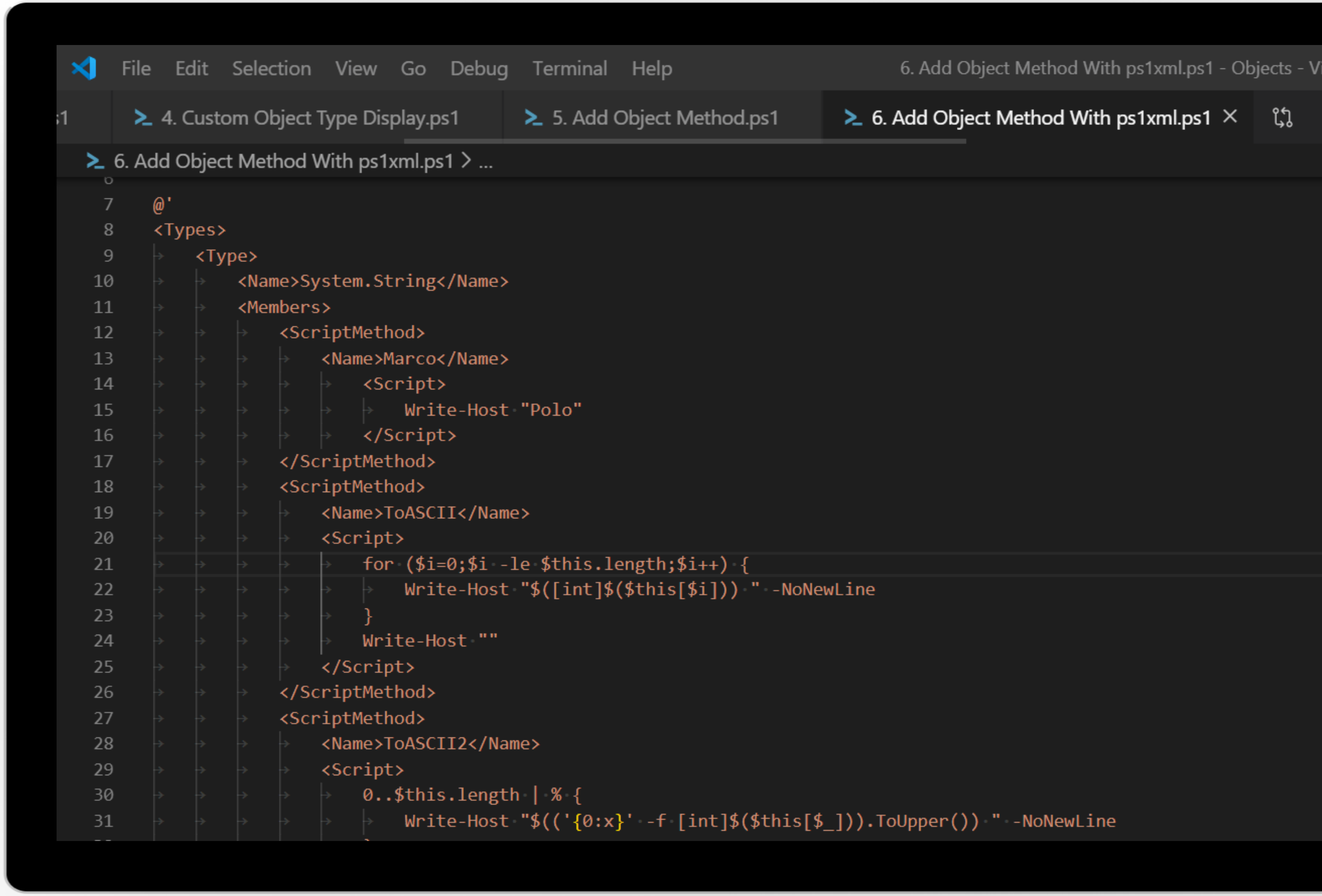
  <# .Synopsis, .Description, .Example, .Outputs #>

  # .ExternalHelp <XML-file-name>

Release Pipeline

- Source control with Git and GitHub or Azure DevOps
- Build scripts with psake
- Tests with Pester and PSScriptAnalyzer
- Publish to the PowerShell Gallery (or your private gallery) with PSDeploy

Demos


File
Edit
Selection
View
Go
Debug
Terminal
Help
6. Add Object Method With ps1xml.ps1 - Objects - V
4. Custom Object Type Display.ps1
5. Add Object Method.ps1
6. Add Object Method With ps1xml.ps1
6. Add Object Method With ps1xml.ps1 > ...

```
@'
<Types>
    <Type>
        <Name>System.String</Name>
        <Members>
            <ScriptMethod>
                <Name>Marco</Name>
                    <Script>
                        Write-Host "Polo"
                    </Script>
            </ScriptMethod>
            <ScriptMethod>
                <Name>ToASCII</Name>
                <Script>
                    for ($i=0;$i -le $this.length;$i++) {
                        Write-Host "$([int]$($this[$i])) " -NoNewLine
                    }
                    Write-Host ""
                </Script>
            </ScriptMethod>
            <ScriptMethod>
                <Name>ToASCII2</Name>
                <Script>
                    0..$this.length | % {
                        Write-Host "$(('{0:x}' -f [int]$($this[$_])).ToUpper()) " -NoNewLine
```

References

About Modules
https://docs.microsoft.com/en-us/powershell/module/microsoft.powershell.core/about/about_modules

How to Write a Module Manifest
https://docs.microsoft.com/en-us/previous-versions//dd878297(v=vs.85)

The Unofficial PowerShell Best Practices and Style Guide
https://github.com/PoshCode/PowerShellPracticeAndStyle

The Release Pipeline Model
https://docs.microsoft.com/en-us/powershell/scripting/dsc/further-reading/whitepapers

Thank you