PowerShell Desired State Configuration (DSC)

Resource Overview

Visual Studio ALM Rangers

1 Getting Ready

- DSC Resource Kit aka.ms/dscRK
- DSC Guidance aka.ms/dscGuide
- PowerShell.org location aka.ms/dscPso
- PowerShell.org eBook aka.ms/dscPsoBook

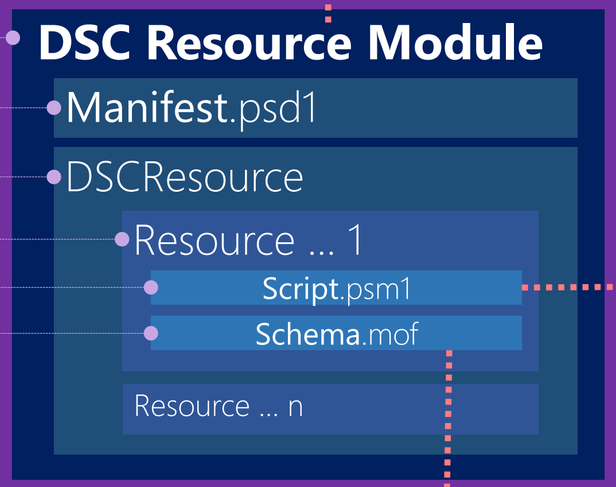
2 Architecture

- Custom module folder
  - **Manifest** (.psd1)
    - DSCResource sub-folder
      - Resource sub folder
        - **Script** module (.psm1)
        - **Schema** (.mof)
- **Resource Designer** creates the structure for you!

```
@{
# Version number of this module.
ModuleVersion = '2.1'

# ID used to uniquely identify this module
GUID = 'e6647cc3-ce9c-4c86-9eb8-2ee8919bf358'

# Author of this module
Author = 'Microsoft Corporation'

# Company or vendor of this module
CompanyName = 'Microsoft Corporation'

# Copyright statement for this module
Copyright = '(c) 2013 Microsoft Corporation. All rights reserved.'

# Description of the functionality provided by this module
Description = 'Module with DSC Resources for Networking area'

# Minimum version of the Windows PowerShell engine required by this module
PowerShellVersion = '4.0'

# Minimum version of the common language runtime (CLR) required by this module
CLRVersion = '4.0'

# Functions to export from this module
FunctionsToExport = '*'

# Cmdlets to export from this module
CmdletsToExport = '*'
}
```

experimental

Example: xNetworking.psd1



Example: MSFT_xFirewall.psm1

Example: MSFT_xFirewall.Schema.mof

3 Dev process

1. **Planning** the resource
   - Define self-config information
2. **Define** properties
   - Use Resource Designer aka.ms/dscRD
3. **Program** resource
   - Get-TargetResource
   - Set-TargetResource
   - Test-TargetResource
4. **Test** & Deploy aka.ms/dscTP

```
$ipaddress = New-DscResourceCategory -Name IPAddress
                                     -Type String
                                     -Attribute Key
…

New-DscResource -Name xIPAddress -properties $ipaddress, …
                -Path c:\temp -FriendlyName xIPAddress
```

generates mof

```
[ClassVersion("1.0.0"), FriendlyName("xIPAddress")]
class MSFT_xIPAddress : OMI_BaseResource
{
  [Key] string IPAddress;
  [Key] string InterfaceAlias;
  [write] string DefaultGateway;
  [Write] uint32 SubnetMask;
  [Write,ValueMap{"IPv4", "IPv6"},Values{"IPv4", "IPv6"}] string AddressFamily;
};
```

Well structured PowerShell Script

```
<#############################################################################
#  MSDSCPack_IPAddress : DSC Resource that will set/test/get the current IP
#  Address, by accepting values among those given in MSDSCPack_IPAddress.schema.mof
#############################################################################>

#############################################################################
# The Get-TargetResource cmdlet.
# This function gets the present list of IP Adr DSC Resource schema variables on system
#############################################################################
function Get-TargetResource
{
    param
    (
        [Parameter(Mandatory)]
        [ValidateNotNullOrEmpty()]
        [String]$IPAddress,

        [Parameter(Mandatory)]
        [ValidateNotNullOrEmpty()]
        [String]$InterfaceAlias,

        [Int]$SubnetMask = 16,

        [ValidateNotNullOrEmpty()]
        [String]$DefaultGateway,

        [ValidateSet("IPv4", "IPv6")]
        [String]$AddressFamily = "IPv4"
    )
    $returnValue = @{
        IPAddress = [System.String]::Join(", ",
        (Get-NetIPAddress -InterfaceAlias $InterfaceAlias -AddressFamily $AddressFamily).IPAddress)
        SubnetMask = $SubnetMask
        DefaultGateway = $DefaultGateway
        AddressFamily = $AddressFamily
        InterfaceAlias=$InterfaceAlias
    }
    $returnValue
}

#############################################################################
# The Set-TargetResource cmdlet.
# This function will set a new IP Address in the current node
#############################################################################
function Set-TargetResource
{
    param
    (
        #IP Address that has to be set
        [Parameter(Mandatory)]
        [ValidateNotNullOrEmpty()]
        [String]$IPAddress,

        [Parameter(Mandatory)]
        [ValidateNotNullOrEmpty()]
        [String]$InterfaceAlias,

        [Int]$SubnetMask,

        [ValidateNotNullOrEmpty()]
        [String]$DefaultGateway,

        [ValidateSet("IPv4", "IPv6")]
        [String]$AddressFamily = "IPv4"
    )
    ValidateProperties @PSBoundParameters -Apply
}

#############################################################################
# The Test-TargetResource cmdlet.
# This will test if the given IP Address is among the current node's IP Address collection
#############################################################################
function Test-TargetResource
{
    param
    (
        [Parameter(Mandatory)]
        [ValidateNotNullOrEmpty()]
        [String]$IPAddress,

        [Parameter(Mandatory)]
        [ValidateNotNullOrEmpty()]
        [String]$InterfaceAlias,

        [Int]$SubnetMask,

        [ValidateNotNullOrEmpty()]
        [String]$DefaultGateway,

        [ValidateSet("IPv4", "IPv6")]
        [String]$AddressFamily = "IPv4"
    )
    ValidateProperties @PSBoundParameters
}

#############################################################################
#  Helper function that validates the IP Address properties. If the switch parameter
# "Apply" is set, then it will set the properties after a test
#############################################################################
…

#  FUNCTIONS TO BE EXPORTED
Export-ModuleMember -function Get-TargetResource, Set-TargetResource, Test-TargetResource
```