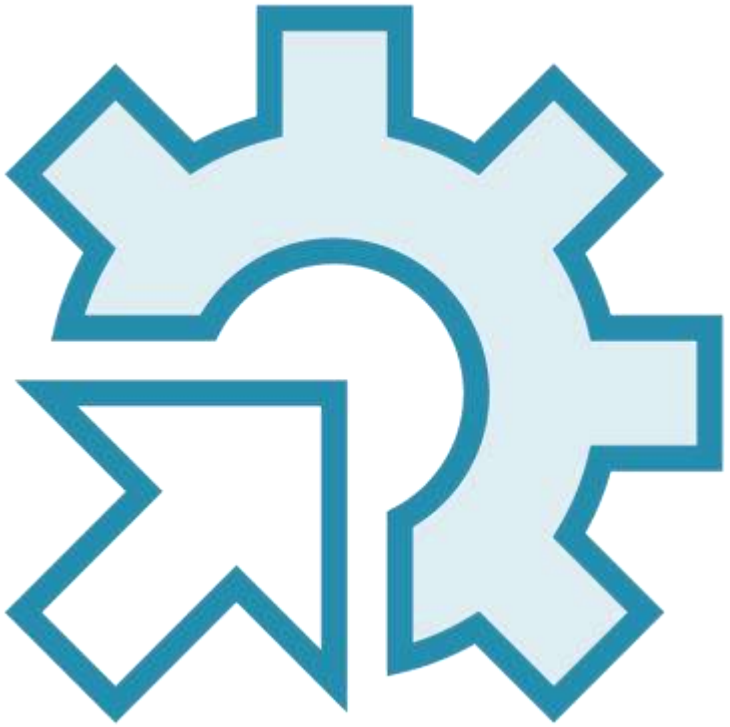
Creating Custom DSC Resources

Jeff Hicks
Author/Teacher

@jeffhicks | https://jdhitsolutions.com/blog

Custom DSC Resources



There is a huge library of DSC resources

But there may be a gap

- No resource for a configuration you need
- Want to support a custom applications

You can create your own DSC resource

- Use the Script resource
- Deploy as a module

DSC Resources

Test

Set

Get

Implemented with PowerShell Code

```
Configuration RSATConfig {

    Import-DSCresource -ModuleName "PSDesiredStateConfiguration" -ModuleVersion "1.1"

    Node Win10 {
        Script RSAT { ... }
 }
}
```

Custom Resources

Use the Script resource
Easy way to prototype
Built-in resource
Doesn’t require a module
Use for simple configurations

```
Node Win10 {
    Script RSAT {
        TestScript = {
            #return true or false
            }
...
```

Custom Resources

Requires a TestScript entry
Scriptblock that returns True or False

```
Node Win10 {
    Script RSAT {
        TestScript = {
            #return true or false
            }
        GetScript = {
             #get value code
            @{Result="result information"}
            }
...
```

Custom Resources

Requires a GetScript entry
Scriptblock that returns a hashtable
Result key and a string value

```
Node Win10 {
    Script RSAT {
        TestScript = {
            #return true or false
            }
        GetScript = {
             #get value code
            @{Result="result information"}
            }
         SetScript = {
                #implementing code
            }
    }
```

Custom Resources

Requires a SetScript entry
Scriptblock that implements the configuration through PowerShell

Demo

Using the Script DSC Resource

Custom DSC Resource Module

Create a traditional PowerShell Module

- ISE: Use the DSC Resource Provider (simple)
- VSCode should have snippets too

Define module functions

- Get-TargetResource
- Set-TargetResource
- Test-TargetResource
- Define helper functions as necessary

Define a module manifest

- Export the DSC Resource

Custom DSC Resource Module

Also requires a .Schema.Mof file

```
C:\Program Files\WindowsPowerShell\Modules\CompanyRSAT
\--CompanyRSAT.psd1
+--DSCResources
|  +--CompanyRSAT
|     \--CompanyRSAT.psm1
|     \--CompanyRSAT.schema.mof
```

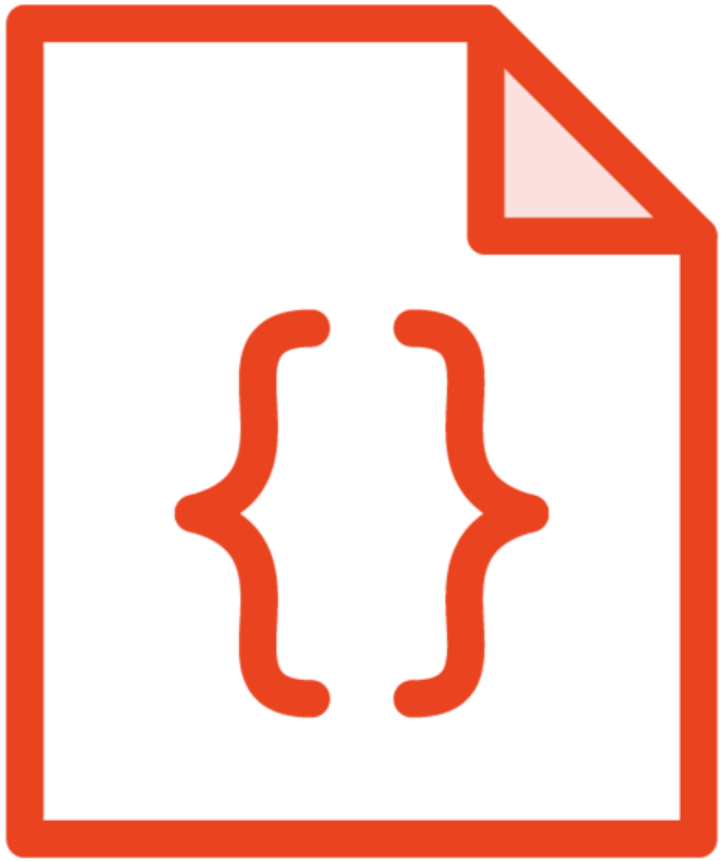
Custom DSC Resource Module

Requires a specify folder structure
The module folder, .psm1 file, and .mof files share resource name

Demo

Creating a MOF-based custom resource

Class-Based DSC Resource Module



Define the resource as a PowerShell class

- Nodes must be running PowerShell 5.0 or later

Does not require a schema MOF

Functions become Methods

- Get()
- Test()
- Set()

Implement with helper functions

This is an advanced PowerShell scripting topic

```
enum Ensure {
    Absent
    Present
}

[DscResource()]
class myClass {
 [DscProperty(Key)]
 [string]$Name

 [DscProperty(Mandatory)]
 [Ensure]$Ensure

 [DscProperty()]
 [string]$Size

 [DscProperty(NotConfigurable)]
 [datetime]$InstallDate
```

- Define enumerations

- [DscResource()] indicates the class is a DSC resource.
- A DSC resource must define at least one mandatory key property
- Mandatory indicates the property is required and DSC will guarantee it is set

- Optional parameter

- NotConfigurable properties return additional information about the state of the resource
- These properties are only used by the Get() method and cannot be set

```
 [MyClass] Get()  {
        $this = #some code
        # Return this instance or
construct a new instance.
        return $this
    }
 [bool] Test()     {
        $result = #some code to get
$True or $False
        return $result
    }
 [void] Set() {
      #some code to set the state
    }
```

- Gets the resource's current state
- You must use Return
- Test if the resource is in the desired state
- Set the resource’s desired state
- You must specify the output type of each method

Demo

Creating a Class-Based Custom Resource