Hands On Lab Instructor Notes

- Estimated duration: 3-4 Hours
- Student computer requirements - see next slide
- Students should have basic experience with deploying Azure resources via the Portal and an understanding of PowerShell.

Student PC Requirements

- Visual Studio Code https://code.visualstudio.com/
- Visual Studio Code Extensions:
  - Azure Resource Manager Tools
  - PowerShell Extension
- Postman
- Owner or Contributor access to an Azure subscription
- Azure SDK 2.9.+
- Git client
- Latest Azure PowerShell Cmdlets
- https://azure.microsoft.com/en-us/downloads/
- Ensure you reboot after installing the SDK or Azure PowerShell will not work correctly
- Internet access

Microsoft

Azure Automation using Runbooks

Kevin Hilscher
Azure Cloud Solution Architect
kevin.hilscher@microsoft.com

Objectives

- Azure Automation Overview
- Understand Azure Automation concepts
  - Runbooks
  - Shared Assets
- PowerShell 101 Review
- Author Runbooks using PowerShell

Azure Automation Capabilities

Process Automation

Orchestrate processes using graphical, PowerShell, and Python runbooks

Configuration Management

Collect inventory

Track changes

Configure desired state

Update Management

Assess compliance

Schedule update installation

Shared capabilities

Role based access control

Secure, global store for variables, credentials, certificates, connections

Flexible scheduling

Shared modules

Source control support

Auditing

Tags

Heterogenous

Windows & Linux

Azure and on-premises

Automation Use Cases

- Automate Configuration Management (CM) Activities
  - Run PowerShell DSC after a change is detected to ensure VM is configured to desired state.
- Automate Alert Responses
  - OMS fires a VM Disk Full alert which runs an Automation Runbook to delete temp files in VM.
  - Azure Service Health can run an Automation Runbook if a service health event is triggered.
- Protect
  - Quarantine VM if security alert is raised.
- Automatically Stop and Start Resources
  - Save money by shutting down resources on evenings or weekends.

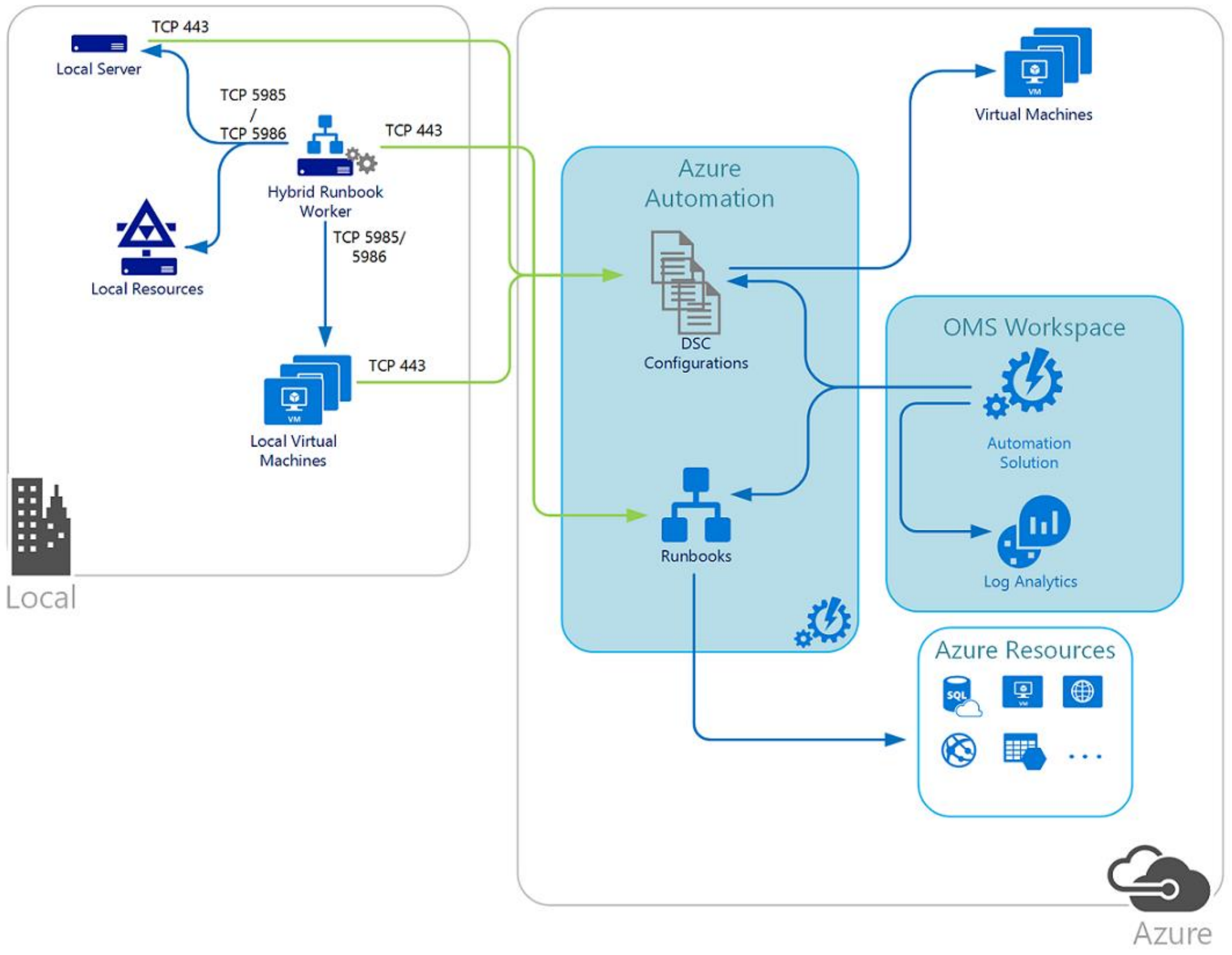
Browse the Automation gallery for runbooks to quickly get started

Automation Accounts

- Azure Automation is a service in which you host runbooks to automate processes.
- You can use Desired State Configuration (DSC) in Azure, other cloud services, or in an on-premises environment to manage configuration changes to Windows and Linux systems.
- Entities in your Automation account, including runbooks, assets, and Run As accounts, are isolated from other Automation accounts in your subscription, and from other subscriptions.
- Automation resources for each Automation account are associated with a single Azure region.
- However, you can use Automation accounts to manage all the resources in your subscription

Architecture

- Log Analytics Workspace
- Automation solution installed and configured in your Log Analytics workspace
- Automation Account
- Hybrid Runbook Worker for on-premise automation



Hybrid Runbook Worker

- Allow runbooks to automate on-premise resources.
- Single or multiple agents in a group, for HA.
- Agent needs outbound port 443 access.
- Min hardware requirements. Linux and Windows.
- Runbook authentication options:
  - Credential asset
    - domain\username
    - username@domain
  - Certificate asset

Pricing (Retail)

- https://azure.microsoft.com/en-us/pricing/details/automation/
- https://azure.microsoft.com/en-us/pricing/details/log-analytics/

Demo

Automation using Runbooks

Runbooks

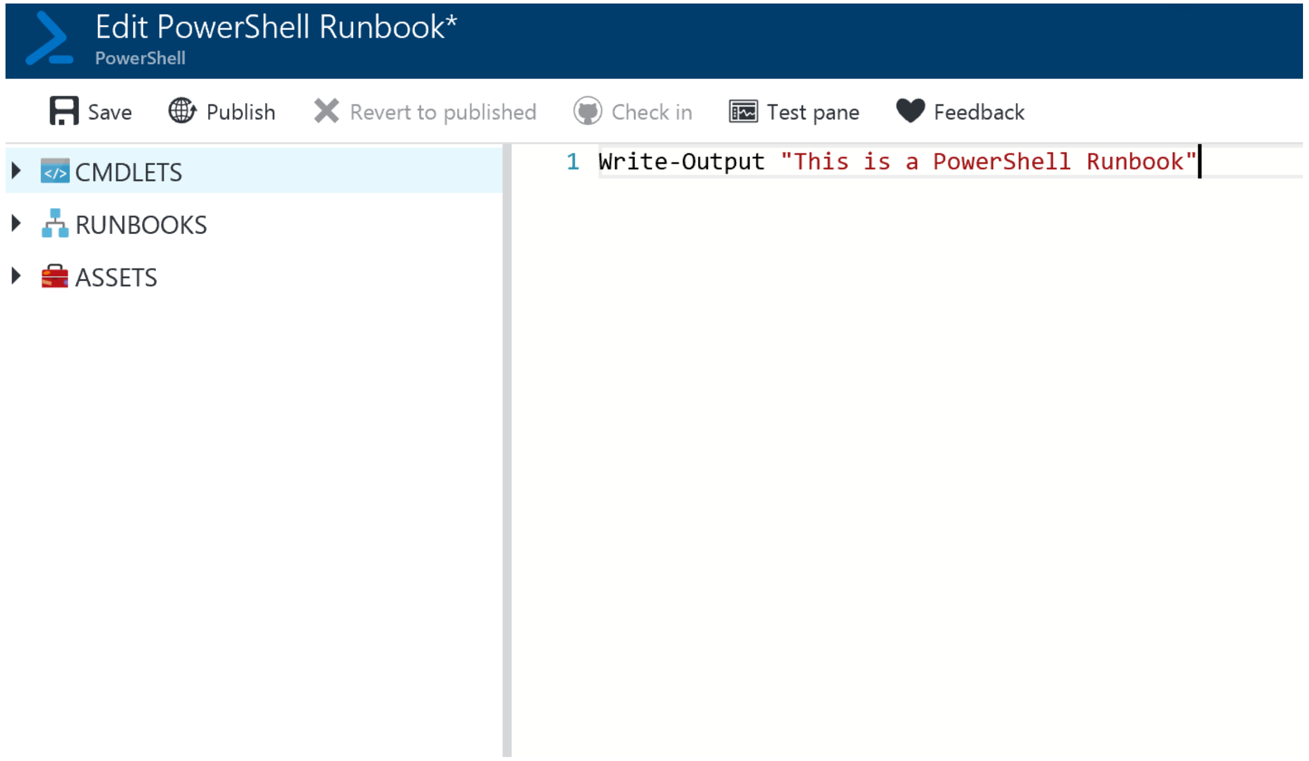
- Types
  - PowerShell
  - PowerShell Workflow (Hint: Do you need checkpoints or parallel processing?)
  - Graphical
  - Graphical PowerShell Workflow (Hint: Do you need checkpoints or parallel processing?)
  - Python
- Create New
- Import from file
- Import from Runbook Gallery
- Viewable without editing
- Exportable

Authoring a Runbook – Which option?

- Recommend you always choose PowerShell
- Graphical runbooks are created and edited with the graphical editor in the Azure portal.
  - You can export them to a file and then import them into another automation account, but you cannot create or edit them with another tool.
  - Graphical runbooks generate PowerShell code, but you can't directly view or modify the code.
- PowerShell runbooks are based on Windows PowerShell. You directly edit the code of the runbook using the text editor in the Azure portal.
  - You can also use any offline text editor and import the runbook into Azure Automation.

Runbooks

- PowerShell
- Limitations
  - Must be familiar with PowerShell
  - No Parallel Execution
  - No Checkpoints
  - Child Runbook uses Start-AzureAutomationRunbook

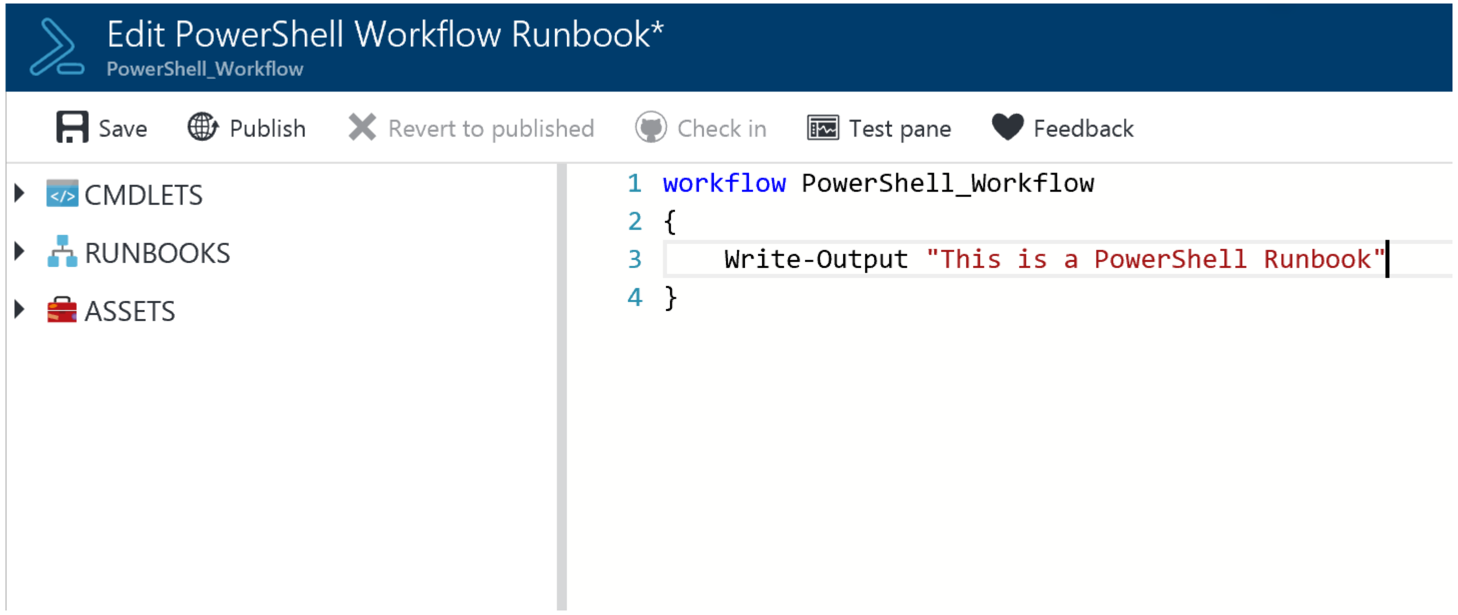


```
Write-Output "This is a PowerShell Runbook"
```

Runbooks

- PowerShell Workflow
  - Text Runbook
  - Based on Windows PowerShell Workflow
  - Use checkpoints
  - Use Parallel processing
  - Can use other Workflow Runbooks as Child Runbooks

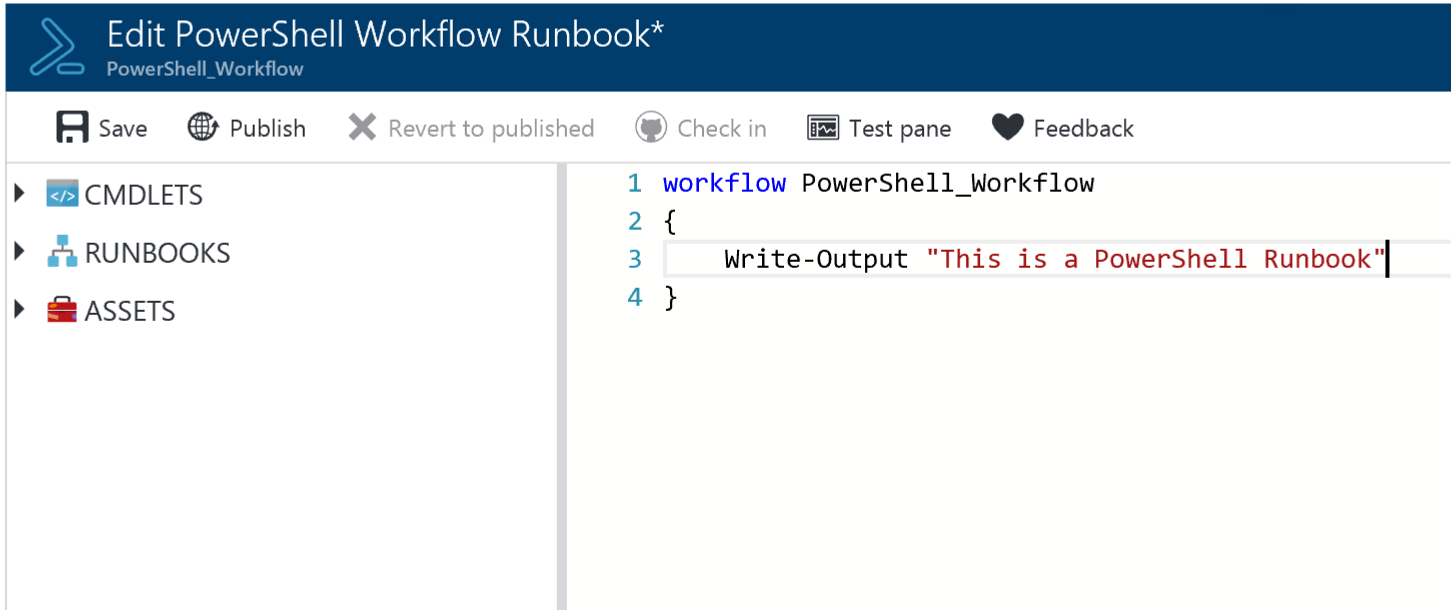
```
ForEach -Parallel ($<item> in $<collection>)
{
  <Activity1>
  <Activity2>
}
<Activity3>
```



```
workflow PowerShell_Workflow
{
    Write-Output "This is a PowerShell Runbook"
}
```

Runbooks

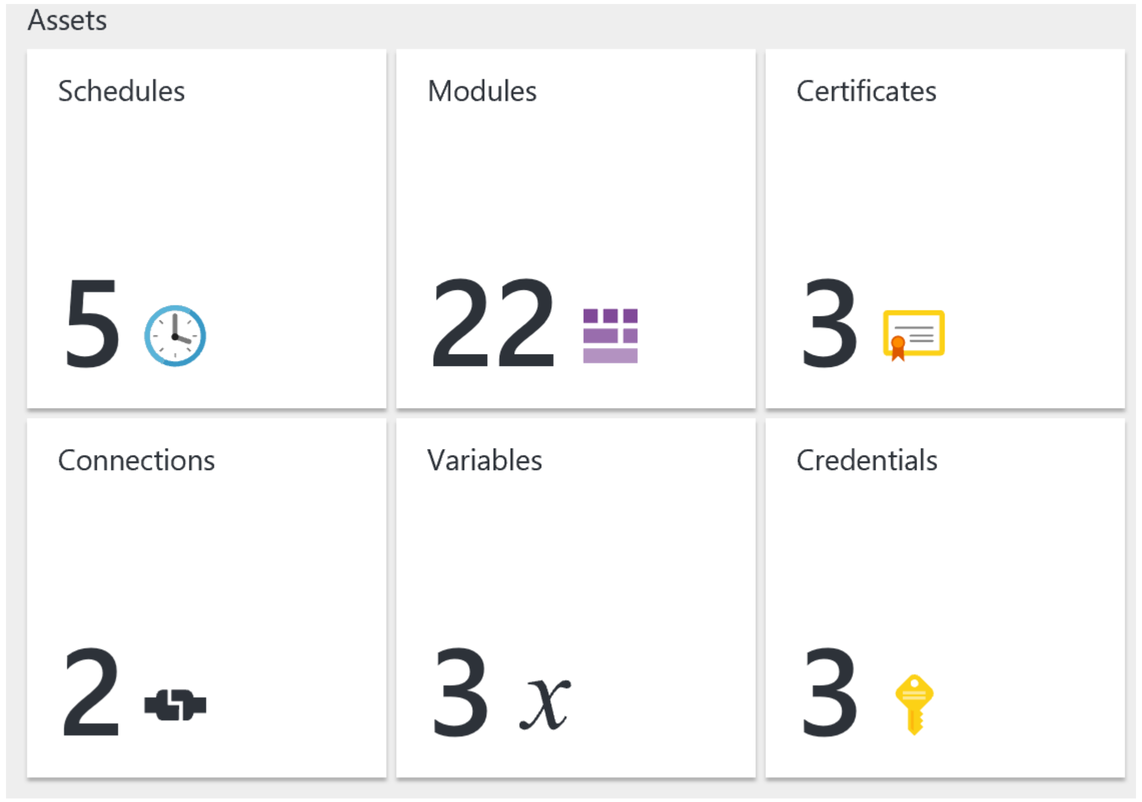
- PowerShell Workflow
- Limitations
  - Must be familiar with PowerShell Workflow
  - Extra complexity e.g. deserialized objects
  - Takes longer than PowerShell
  - Calling PowerShell Child Runbooks need to use Start-AzureAutomationRunbook



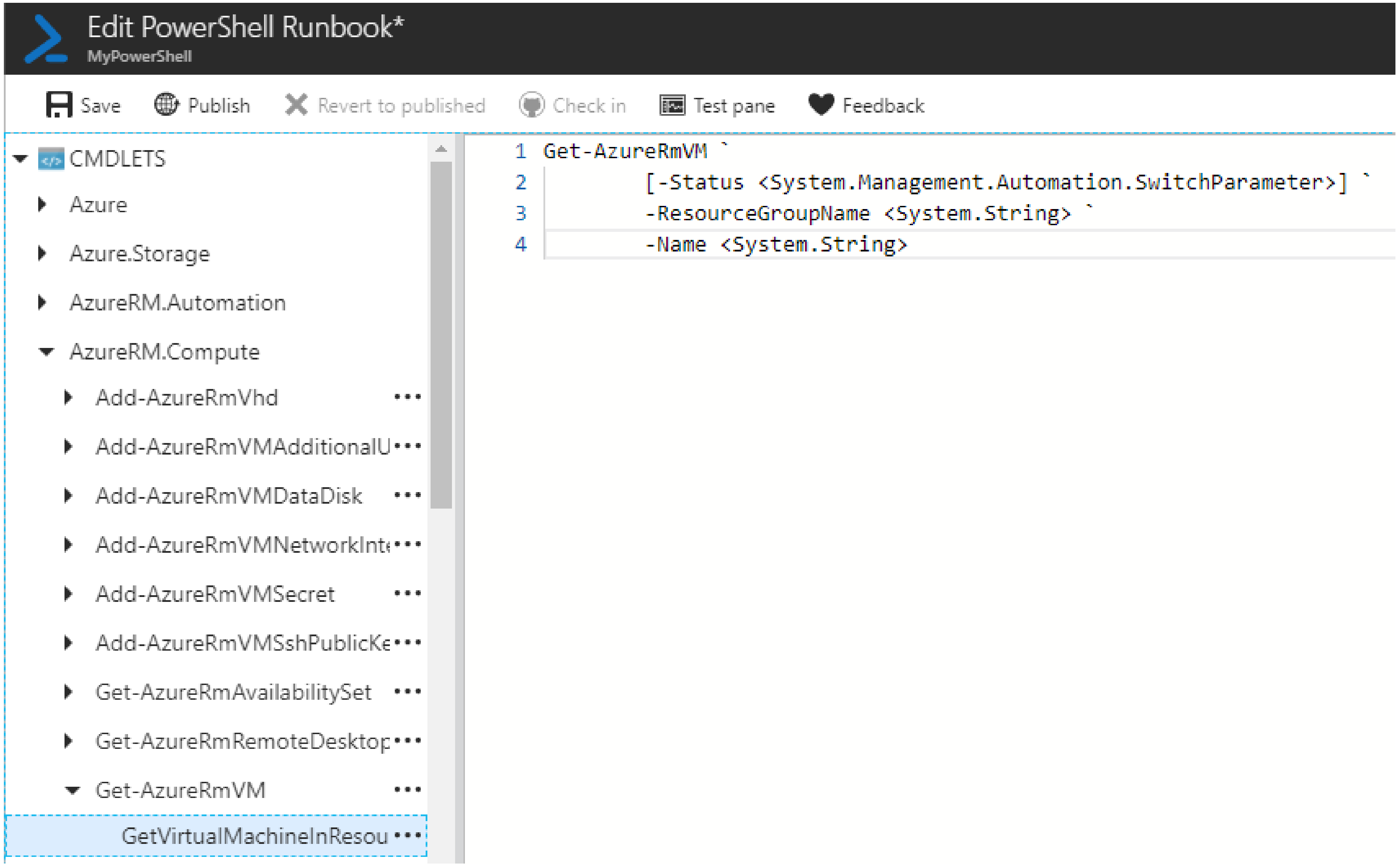
```
workflow PowerShell_Workflow
{
    Write-Output "This is a PowerShell Runbook"
}
```

Shared Assets

- Modules
- Schedules
- Certificates
- Credentials
- Variables
- Connections



Cmdlets



```
Get-AzureRmVM `
        [-Status <System.Management.Automation.SwitchParameter>] `
        -ResourceGroupName <System.String> `
        -Name <System.String>
```

Run As Accounts

- Creates a service principal in Azure AD.
- Creates a certificate.
- Assigns the Contributor Role-Based Access Control (RBAC), which manages Azure Resource Manager resources by using runbooks.

Demo – Authoring a PowerShell Runbook

Authoring Tools

- Command line
- Notepad/Textpad/…any favorite editor
- PowerShell ISE (Deprecated)
- Visual Studio Code + PowerShell extension
  - Intellisense
  - Save your scripts with **.ps1** extension which invokes integrated PS
  - **F5** runs entire script with debugging
  - **Ctrl + F5** runs entire script without debugging
  - Select a line and press **F8** to run a single line

Demo – Using Webhooks

Calling a webhook from Postman with a Body

```
[CmdletBinding()]
Param
([object]$WebhookData) #this parameter name needs to be called WebHookData otherwise the webhook does not work as expected.
$VerbosePreference = 'continue'

#region Verify if Runbook is started from Webhook.

# If runbook was called from Webhook, WebhookData will not be null.
if ($WebHookData){

    # Collect properties of WebhookData
    $WebhookName     =     $WebHookData.WebhookName
    $WebhookHeaders  =     $WebHookData.RequestHeader
    $WebhookBody     =     $WebHookData.RequestBody

    # Collect individual headers. Input converted from JSON.
    $From = $WebhookHeaders.From
    $Input = (ConvertFrom-Json -InputObject $WebhookBody)
    Write-Verbose "WebhookBody: $Input"
    Write-Output -InputObject ('Runbook started from webhook {0} by {1}.' -f $WebhookName, $From)
}
else
{
   Write-Error -Message 'Runbook was not started from Webhook' -ErrorAction stop
}
#endregion

#region Main
$FirstName = $Input.FirstName
$LastName = $Input.LastName

 Write-Output -InputObject ('Hello {0} {1}.' -f $FirstName, $LastName)
 #endregion
```

PowerShell 101

Help

- Autocomplete
  - Example:
    - `Add-AzureRmaccount -<hit tab key>`
- Get-Help <cmdlet>
  - Example:
    - `Get-Help Add-AzureRMAccount`
- Update Help Documentation
  - Example:
    - `Update-Help`

Logging into Azure using PowerShell

- The ARM cmdlets **Login-AzureRMAccount** and **Add-AzureRMAccount** are synonymous thanks to PowerShell's command alias system.
- **Login-AzureRMAccount** aliases back to **Add-AzureRMAccount**
- Azure Automation uses **Add-AzureRMAccount** by default.

Logging into Azure using PowerShell

```
#Login using AAD
Login-AzureRMAccount
```

```
# Authenticate to Azure if running from Azure Automation
$ServicePrincipalConnection = Get-AutomationConnection -Name
"AzureRunAsConnection"
Add-AzureRmAccount `
    -ServicePrincipal `
    -TenantId $ServicePrincipalConnection.TenantId `
    -ApplicationId $ServicePrincipalConnection.ApplicationId `
    -CertificateThumbprint $ServicePrincipalConnection.CertificateThumbprint
| Write-Verbose
```

Set your Subscription context

```
#Show all your subscriptions
Get-AzureRmSubscription

SubscriptionName : sub1
SubscriptionId : 3d935138-40b5-408c-98e9
TenantId : 133f6972-44a7-4037-8eea
State : Enabled
```

```
#Set your subscription context
Get-AzureRMSubscription -SubscriptionName 'sub2' | Set-AzureRMContext
```

Examples

```
#Prompt for a parameter
param (
    [Parameter(Mandatory=$true)]
    [string]
    $adminUsername
)
```

```
#Display output
Write-Output "Hello World"
```

```
#Assign a variable
$var = "string"
```

```
#Interactive user prompts
$age = Read-host "Please enter your age"
```

```
#If Then Else + cmdlet example
$ChkFile = "C:\Windows\explorer.exe"
$FileExists = Test-Path $ChkFile
If ($FileExists -eq $True) { Write-Host "Yippee, explorer.exe exists" }
Else {Write-Host "explorer.exe does not exist" }
```

http://ramblingcookiemonster.github.io/images/Cheat-Sheets/powershell-basic-cheat-sheet2.pdf

Hands on Labs

Exercise 1: Create an Automation Account

1. Create an Azure Automation Account with a Run As account

Note: Don’t link it to an OMS Workspace

Exercise 2: Import a Runbook from the Gallery

- Find a Runbook in the Gallery that lists Resources in your Azure subscription.
- Try several Runbooks…see which you prefer.

Exercise 3: Create a new PowerShell runbook

- Runs using a Service Principal (Run As)
- Runs on a recurring schedule
- Uses variables
- Uses parameters
- Does something interesting…such as creates a Storage Account.

- Important: Copy the Webhook URL before clicking OK.

Exercise 4: Using Webhooks

- Create a PowerShell Runbook that is triggered by a Webhook.
- Demonstrate sending JSON name/value pairs in the HTTP POST body using Postman.

Exercise 5: Deploy a VM using a Runbook

- Write a PowerShell script, or import one from the Gallery, to deploy an Azure VM using an ARM template.
- Hint:
  - "Azure Quickstart Templates"

Wrap Up

- Delete any Resource Groups created.
- Go enjoy a cold beer!

Microsoft