Project Bicep: ARM Templates RELOADED

with a WVD focus

Esther Barthel

@virtuEs_IT

github.com/cognitionit

Microsoft MVP

Freek Berson

@fberson

github.com/fberson

Microsoft MVP


ROADSHOW
April 29th
April 14th
April 7th
February 17th
February 4th
January 29th
Technine User Group
Detron Knowledge event
Dutch DevOps & Github comm.
Ask Wortell Live Stream
Azure Thursday
Azure Lowlands


BE A
NINJA
UNICORN

Agenda

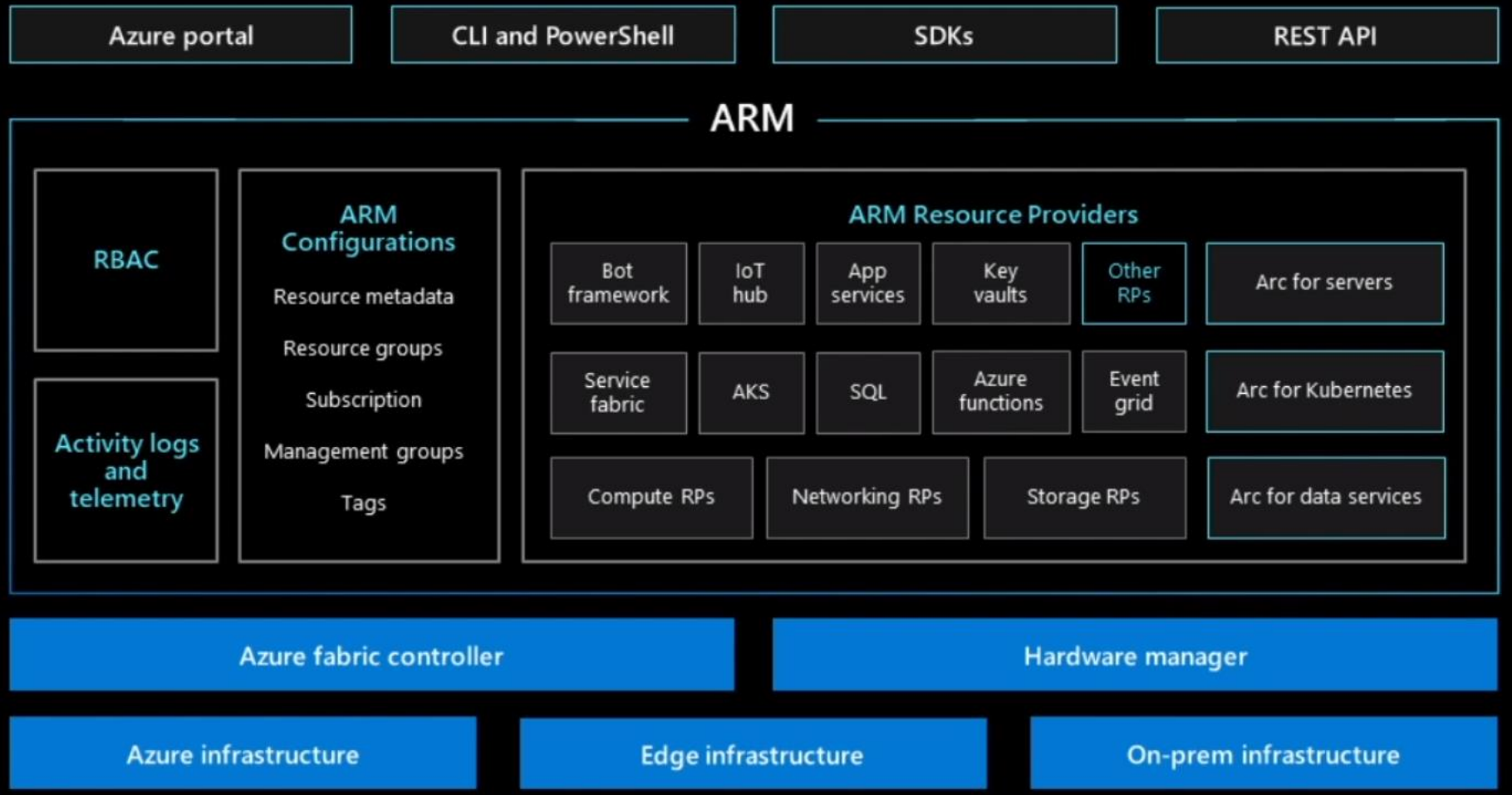
Azure Resource Manager & JSON

Project 'Bicep' architecture

Bicep demo with WVD Focus

Bicep Roadmap & call to actions

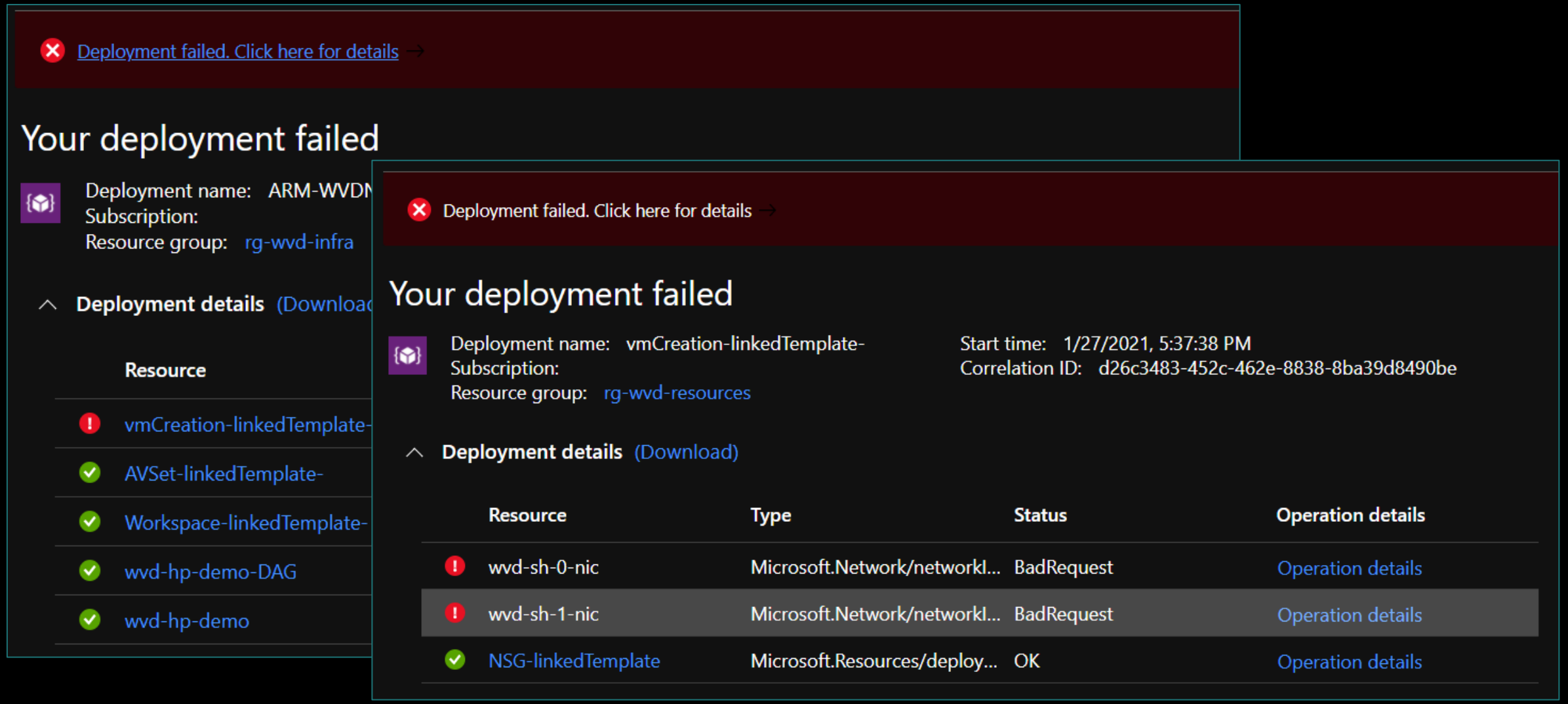
Azure Resource Manager



ARM Template – Resources

- **ARM template documentation**
  https://docs.microsoft.com/en-us/azure/azure-resource-manager/templates/
- **Reference guide:**
  https://docs.microsoft.com/en-us/azure/templates/
- **Learning Path:**
  https://docs.microsoft.com/en-us/learn/paths/deploy-manage-resource-manager-templates/
- **Azure Quickstart Templates:**
  https://azure.microsoft.com/en-us/resources/templates/

ARM Template – Automatic deployments

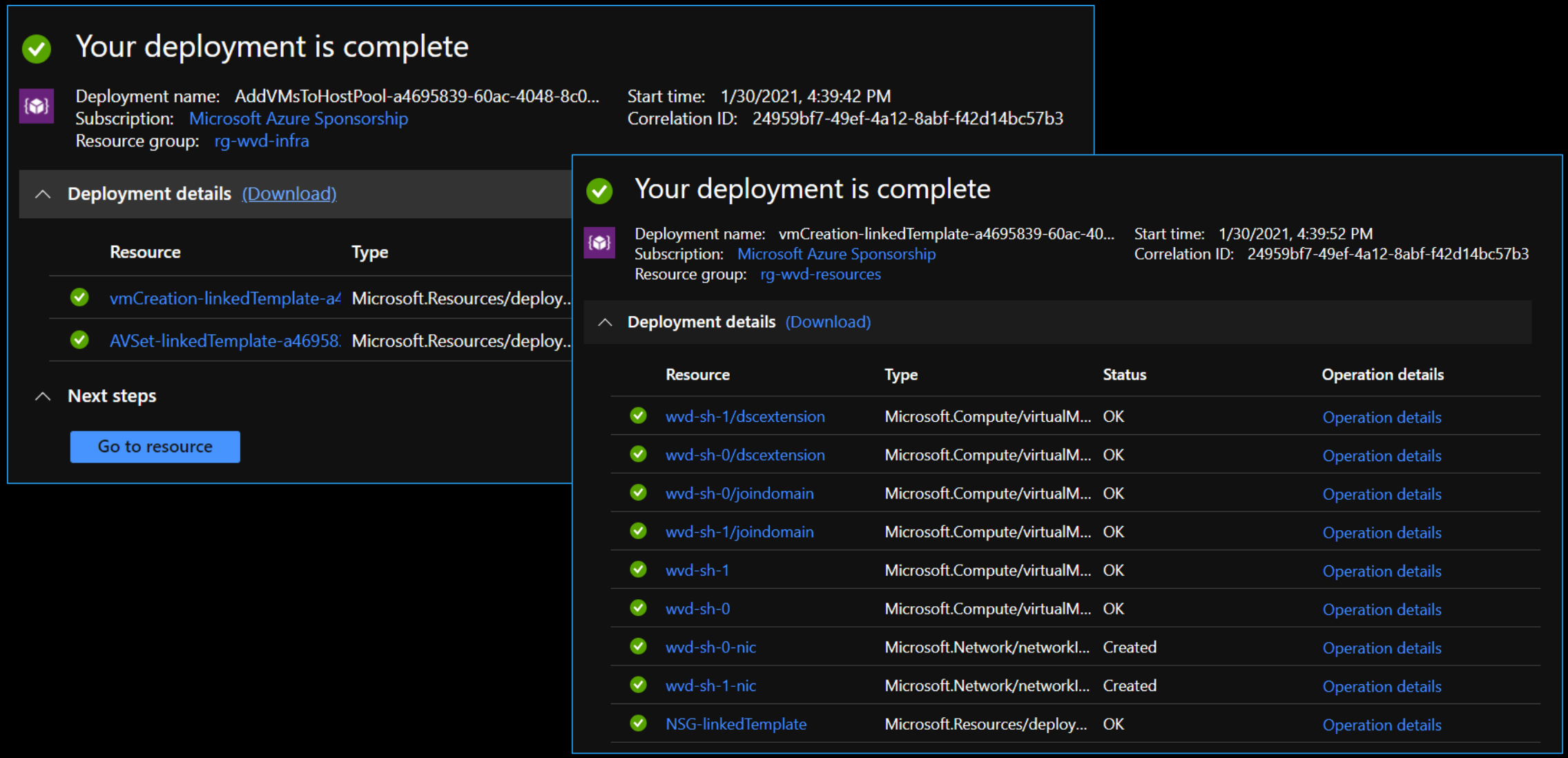


ARM Template – Automatic deployments

```
{
    "$schema": "https://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json#",
    "contentVersion": "1.0.0.0",
    Select or create a parameter file to enable full validation...
    "parameters": {···
    },
    "variables": {···
    },
    "resources": [
        {
            "apiVersion": "2018-05-01",
            "name": "[concat('UpdateHostPool-', parameters('deploymentId'))]",
            "type": "Microsoft.Resources/deployments",
            "resourceGroup": "[parameters('hostpoolResourceGroup')]",
            "condition": "[not(empty(parameters('hostpoolResourceGroup')))]",
            "properties": {···
            }
        },
        {···
        },
        {
            "apiVersion": "2018-05-01",
            "name": "[concat('vmCreation-linkedTemplate-', parameters('deploymentId'))]",
            "resourceGroup": "[parameters('vmResourceGroup')]",
            "dependsOn": [
                "[concat('AVSet-linkedTemplate-', parameters('deploymentId'))]"
            ],
            "type": "Microsoft.Resources/deployments",
            "properties": {···
            }
        }
    ],
    "outputs": {
        "rdshVmNamesObject": {
            "value": "[variables('rdshVmNamesOutput')]",
            "type": "object"
        }
    }
}
```

- 531 lines of code
- complex JSON formatting
- advanced options:
  - nested templates
  - linked templates

ARM Template – Automatic deployments



What is Project ‘Bicep’?

"..Bicep is a Domain Specific Language (DSL) for deploying Azure resources declaratively. It aims to drastically simplify the authoring experience with a cleaner syntax and better support for modularity and code re-use. Bicep is a transparent abstraction over ARM and ARM templates.

Project 'Bicep'

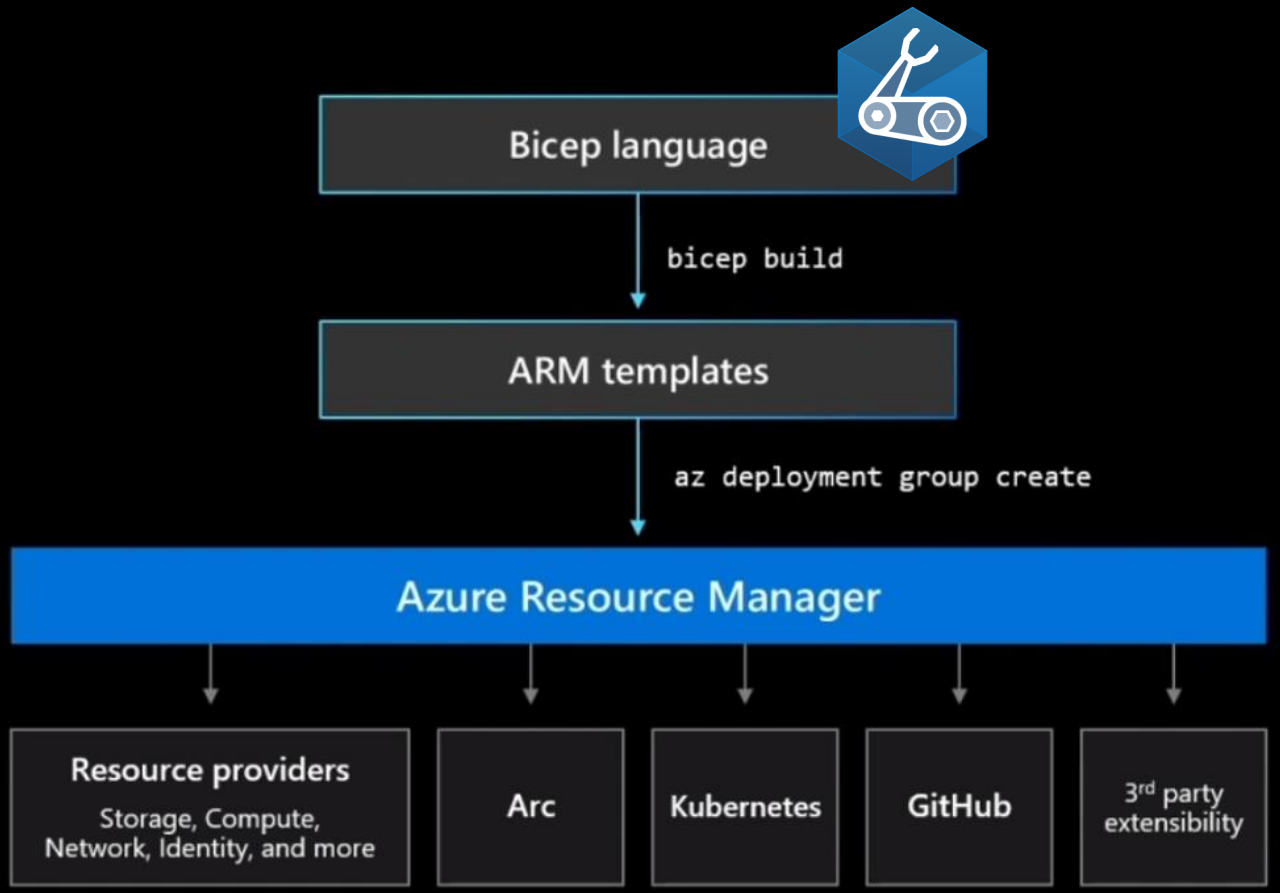
Simple declarative language to provision infrastructure to Azure.

Intuitive
Easy to read and to author

Transpiles to ARM Templates
Leverage ARM template knowledge and investments

Modular
Abstract common blocks of code into reusable parts

Open Source
Transparency and community



How to get started with 'Bicep'?

1. Install the Bicep CLI (required)

```
# Create the install folder
$installPath = "$env:USERPROFILE\.bicep"
$installDir = New-Item -ItemType Directory -Path $installPath -Force
$installDir.Attributes += 'Hidden'
# Fetch the latest Bicep CLI binary
(New-Object Net.WebClient).DownloadFile("https://github.com/Azure/bicep/releases/latest/download/bicep-win-x64.exe", "$installPath\bicep.exe")
# Add bicep to your PATH
$currentPath = (Get-Item -path "HKCU:\Environment" ).GetValue('Path', '', 'DoNotExpandEnvironmentNames')
if (-not $currentPath.Contains("%USERPROFILE%\.bicep")) { setx PATH ($currentPath + ";%USERPROFILE%\.bicep") }
if (-not $env:path.Contains($installPath)) { $env:path += ";$installPath" }
# Verify you can now access the 'bicep' command.
bicep --help
# Done!
```

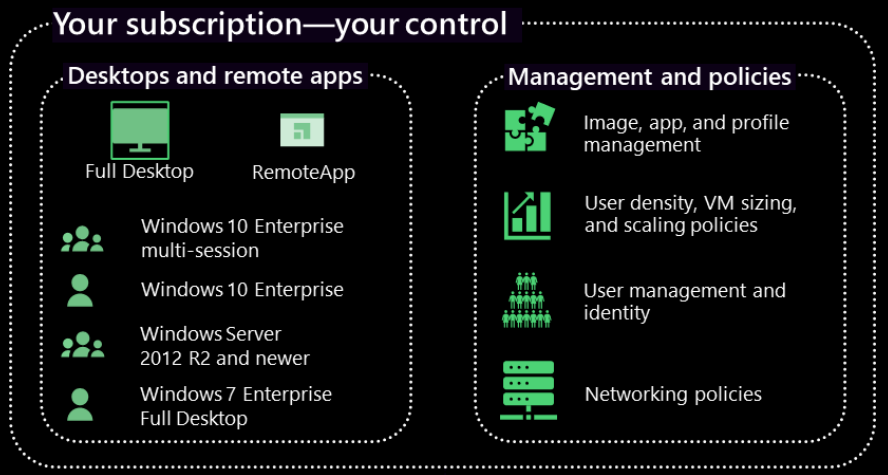
2. Install the Bicep VS Code extension(optional)

```
# Fetch the latest Bicep VSCode extension
$vsixPath = "$env:TEMP\vscode-bicep.vsix"
(New-Object Net.WebClient).DownloadFile("https://github.com/Azure/bicep/releases/latest/download/vscode-bicep.vsix", $vsixPath)
# Install the extension
code --install-extension $vsixPath
# Clean up the file
Remove-Item $vsixPath
# Done!
```

Demo – Putting Bicep into (GitHub) Action

3

Host deployment

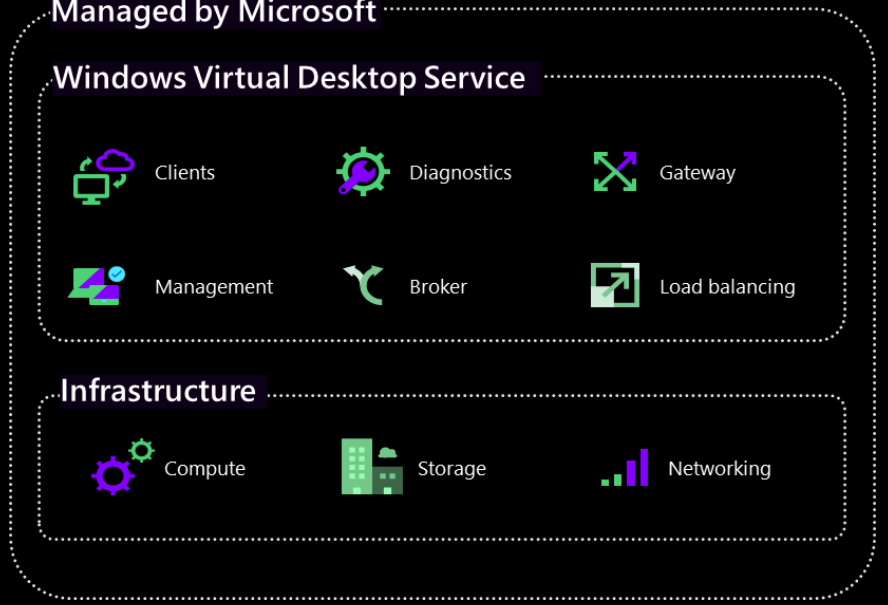


2

Image management

1

WVD backplane deployment



Demo - Putting Bicep into GitHub Action

Search or jump to... / Pull requests Issues Marketplace Explore

cognitionIT / AzureWVD

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

ARM deployment with bicep ARM deployment with bicep

Summary

Jobs

ARM Deployment with bicep file

ARM Deployment with bicep file

succeeded 3 days ago in 6m 30s

- Set up job
- Checkout
- Login via Az module
- ① Prep runner with Bicep prereqs using Azure PowerShell
- ② Deploy ARM Resources with Bicep & Azure PowerShell
- Post Checkout
- Complete job

Deploy local 'Bicep' files

Note: Currently, both Azure CLI and Azure PowerShell can only deploy local Bicep files.
Bicep CLI is needed locally to compile Bicep files to JSON templates before deployment.

Azure CLI v2.20.0+

Azure CLI

```
az deployment group create \
  --name ExampleDeployment \
  --resource-group ExampleGroup \
  --template-file <path-to-template-or-bicep> \
  --parameters storageAccountType=Standard_GRS
```

Note: with Azure CLI v2.20.0+ installed, the Bicep CLI is automatically installed when a command that depends on it is executed.

PowerShell 5.6.0+

Azure PowerShell

```
New-AzResourceGroupDeployment `
  -Name ExampleDeployment `
  -ResourceGroupName ExampleGroup `
  -TemplateFile <path-to-template-or-bicep> `
  -storageAccountType Standard_GRS
```

Note: Azure PowerShell does not have the capability to install the Bicep CLI yet. Azure PowerShell (v5.6.0+) expects that the Bicep CLI is already installed and available on the PATH.

Tip: run Bicep on windows-latest agent

```yaml
# Action = Azure PowerShell: Run inline script
# source: https://github.com/marketplace/actions/azure-powershell-action
- name: Install the min. version Az Module using Azure PowerShell
  uses: azure/powershell@v1
  with:
    inlineScript: |
      ## Add Az PowerShell Module version 5.6.0 to the runner (if not already on the runner)
      $minAzModuleVersion = '5.6.0'
      if(!(Test-Path "C:\Modules\az_$minAzModuleVersion")) {
        Install-Module -Name Az -AllowClobber -Scope CurrentUser -Force
        Save-Module -Path "C:\Modules\az_$minAzModuleVersion" -Name Az -RequiredVersion $minAzModuleVersion -Force
      }
      $env:PSModulePath = "C:\Modules\az_$($minAzModuleVersion);$($env:PSModulePath)"
      # Check installed versions of Az Module
      Get-InstalledModule -Name Az -AllVersions | sort Version -Descending
    azPSVersion: 'latest'
```

ARM Deployment with bicep file

succeeded on Mar 16 in 6m 54s

- Set up job — 24s
- Checkout — 14s
- Login via Az module — 1m 54s
- Prep runner with Bicep prereqs using Azure PowerShell — 3m 56s

```
1   ▶ Run azure/powershell@v1
41  Validating inputs
42  Initializing Az Module
55  ***
56    "Success": "true",
57    "AzVersion": "5.5.0"
58  ***
59  Initializing Az Module Complete
60  Running Az PowerShell Script
66  Bicep CLI version 0.3.1 (d0f5c9b164)
67
97  Version              Name                                Repository           Description
98  -------              ----                                ----------           -----------
99  5.6.0                Az                                  PSGallery            Microsoft Azure PowerShell - Cmdlets to …
100
101 Script execution Complete
```

```yaml
# Action = Azure PowerShell: Run inline script
# source: https://github.com/marketplace/actions/azure-powershell-action
- name: Deploy ARM Resources with Bicep & Azure PowerShell
  uses: azure/powershell@v1
  with:
    inlineScript: |
      # Read the GitHub Actions variables
      [string]$githubWorkspace = "${{GITHUB.WORKSPACE}}"
      # Read the environment variables in PowerShell
      [string]$location = [System.Environment]::GetEnvironmentVariable('LOCATION')
      [string]$bicepFile = [System.Environment]::GetEnvironmentVariable('BICEP_FILE')
      [string]$resourcegroupName = [System.Environment]::GetEnvironmentVariable('RESOURCE_GROUP_NAME')

      $namePostFix = $resourcegroupName.Replace("rg-","")

      ## Create a Template Parameter Object (hashtable)
      $objTemplateParameter = @{
        "location" = "$($location)";
        "workSpaceName" = "ws-$($namePostFix)";
        "hostpoolName" = "hp-$($namePostFix)";
        "appgroupName" = "ag-$($namePostFix)";
        "preferredAppGroupType" = "Desktop";
        "hostPoolType" = "pooled";
        "loadbalancertype" = "DepthFirst";
        "appgroupType" = "Desktop";
      }
      # Location of the bicep file in the local checked-out repo
      $biceptemplateFile = [string]("$($githubWorkspace)" + "\bicep\" + "$($bicepFile)")

      ## Deploy resources based on bicep file for ARM Template
      New-AzResourceGroupDeployment -ResourceGroupName $resourcegroupName -TemplateFile $($biceptemplateFile) -TemplateParameterObject $objTemplateParameter
    azPSVersion: 'latest'
```

ARM Deployment with bicep file

succeeded on Mar 16 in 6m 54s

- Set up job — 24s
- Checkout — 14s
- Login via Az module — 1m 54s
- Prep runner with Bicep prereqs using Azure PowerShell — 3m 56s
- Deploy ARM Resources with Bicep & Azure PowerShell — 23s

```
74  * BICEP FILE: demo.bicep
75  * RESOURCE GROUP NAME: rg-bicepdemo
76  * GITHUB_WORKSPACE: D:\a\AzureWVD\AzureWVD
77  * BICEP TEMPLATE FILE: D:\a\AzureWVD\AzureWVD\bicep\demo.bicep
85  VERBOSE: Performing the operation "Creating Deployment" on target "rg-bicepdemo".
86  VERBOSE: 9:35:24 PM - Template is valid.
87  VERBOSE: 9:35:26 PM - Create template deployment 'demo'
89  VERBOSE: 9:35:31 PM - Resource Microsoft.DesktopVirtualization/hostPools 'hp-bicepdemo' provisioning status is succeeded
91  VERBOSE: 9:35:37 PM - Resource Microsoft.DesktopVirtualization/workspaces 'ws-bicepdemo' provisioning status is succeeded
92  VERBOSE: 9:35:37 PM - Resource Microsoft.DesktopVirtualization/applicationGroups 'ag-bicepdemo' provisioning status is succeeded
94  ResourceGroupName       : rg-bicepdemo
95  OnErrorDeployment       :
96  DeploymentName          : demo
97  CorrelationId           : 3f74503c-3013-47e1-a307-854615ee81a2
98  ProvisioningState       : Succeeded
99  Timestamp               : 3/16/2021 9:35:36 PM
100 Mode                    : Incremental
```

Road map

Current release: CLI version 0.3.255

Or the nightly release for dare devils! ☺

github.com/Azure/bicep/blob/main/docs/installing-nightly.md

v0.1

(aug '20)

Alpha Release available on August 31st

v0.2

(Oct '20)

- VSCode Intellisense
- Support for modules

v0.3

(March '21)

- Loops
- Conditionals
- Decompiler
- Production usage

v0.4

- Quality release
- Learn module Available
- Linter (TTK successor)
- Snippets & resource scaffolding Available*
- Merging ARM Quickstarts & bicep
- IncludeFile() support

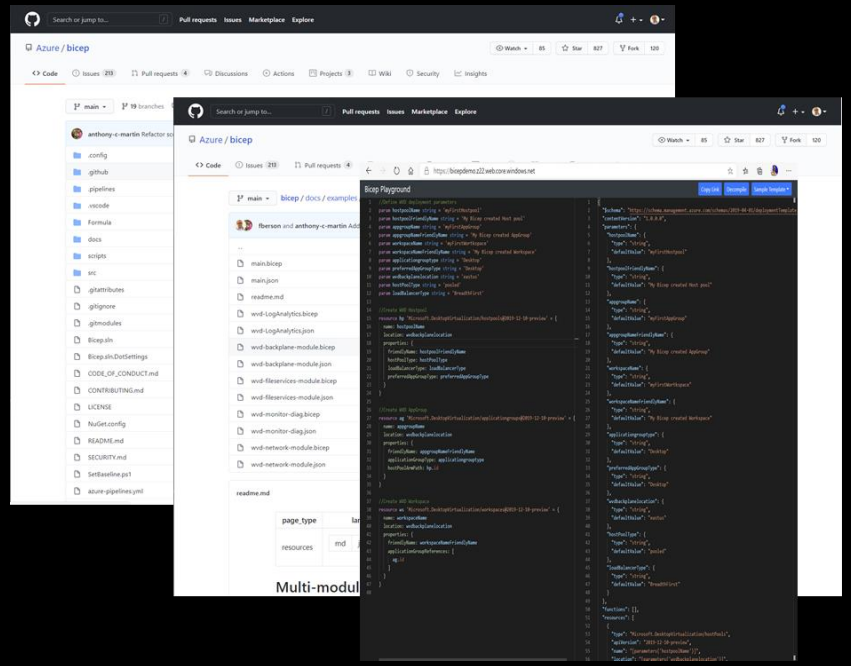
v0.5++

- Module Registry

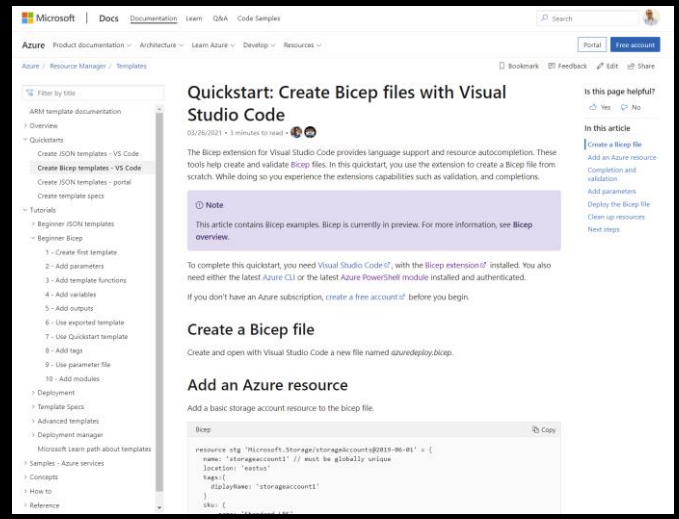
Call to actions:

Install guides, tutorials, example code & playgrounds!

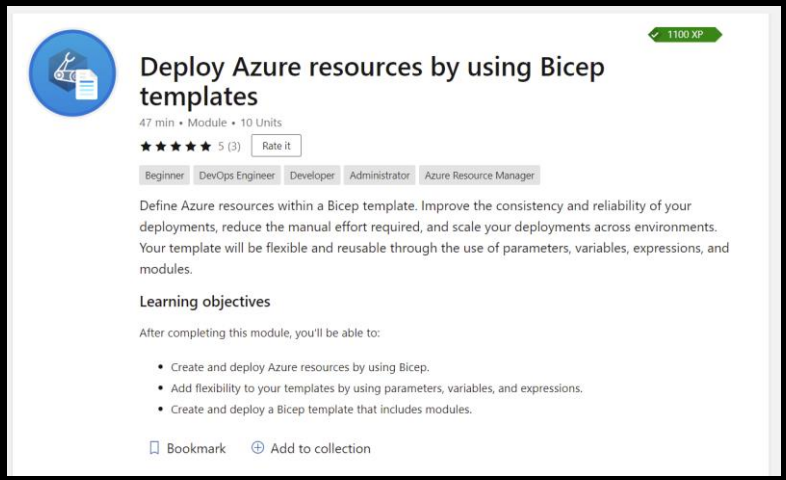
aka.ms/bicep



bit.ly/3ml2FnJ



bit.ly/3wIIuVA



The curious case of the templateHash

Bicep-generated files should include an autogenerated header #800

Closed **snarkywolverine** opened this issue on Nov 3, 2020 · 2 comments

Discussed it at the team meeting today. The consensus appears to have template code generators use the top-level `metadata` property to store this information. This is the proposed schema:

```
"metadata": {
  "_generator": {
    "name": "<name of the code generator>",
    "version": "<version of the code generator>",
    "templateHash": "<template hash>"
  }
}
```

Considerations:

- Discussed using a comment instead of a JSON property. We're not in favor of using meaningful comments due to their fragility and uneven support in JSON libraries across all the relevant platforms.
- Template hash logic should reuse the existing template hash calculation logic that we already have in ARM telemetry and exposed in the API at https://github.com/Azure/azure-rest-api-specs/blob/8cef8014762a839e98f0aeaa57a0bbdb8982d3d4/specification/resources/resource-manager/Microsoft.Resources/stable/2020-10-01/resources.json#L4236
- Template hash calculation should run on the entire content of the template except for the `metadata._generator.templateHash` property. This is technically a breaking change in ARM, but impact should be extremely low.
- Also discussed adding a top-level multi-line comment with text similar to "This file is generated. Do not modify." This should be deferred until we fix bugs in line number handling in the runtime.

Esther Barthel

@virtuEs_IT

github.com/cognitionit

Freek Berson

@fberson

github.com/fberson