Project Bicep: ARM Templates

RELOADED

Esther Barthel

@virtuEs_IT

github.com/cognitionit

Microsoft MVP

Freek Berson

@fberson

github.com/fberson

Microsoft MVP

Agenda

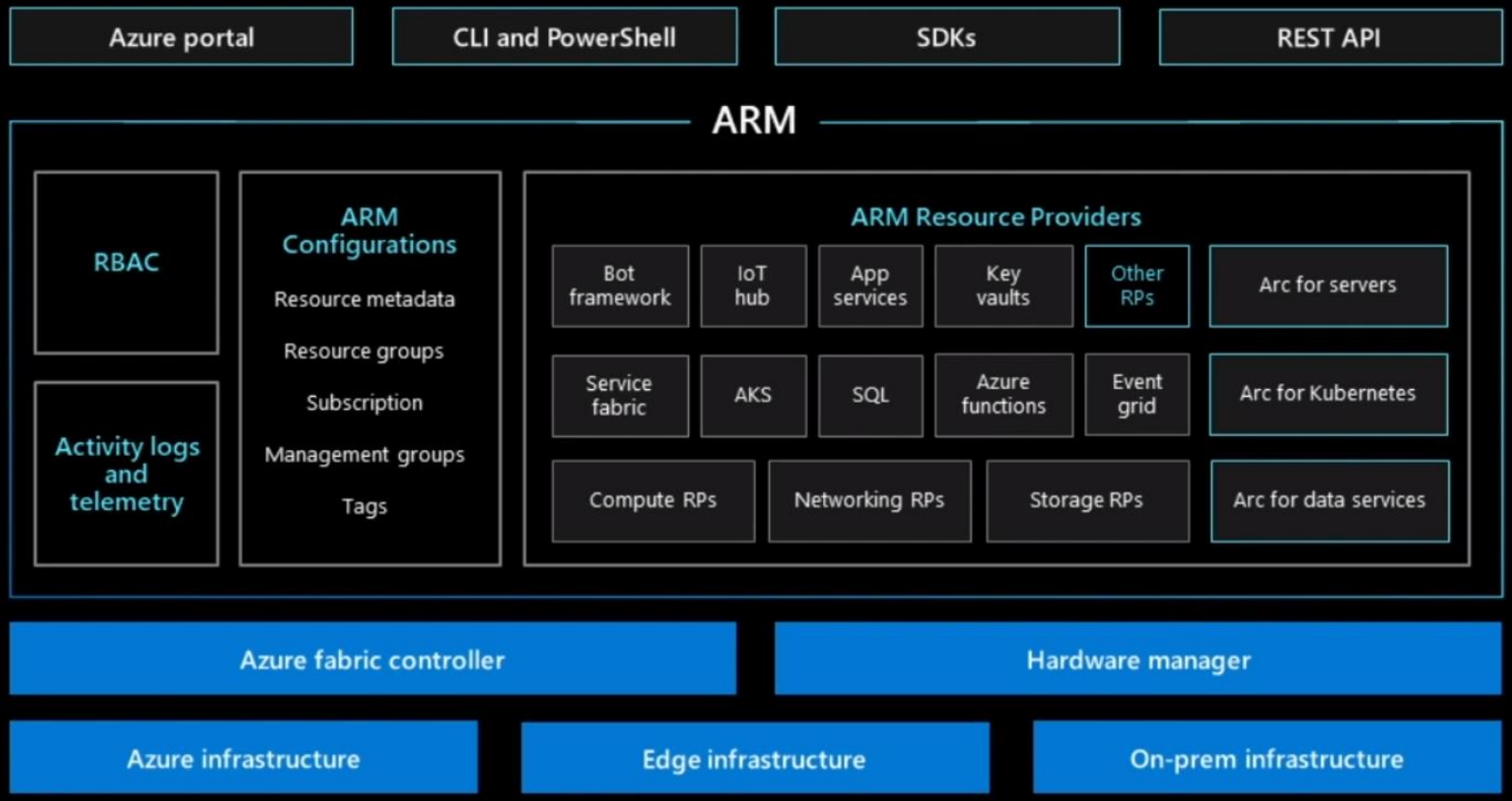
Azure Resource Manager & JSON

Project 'Bicep' architecture

Demos, demos, demos!

Bicep Roadmap & call to action

Azure Resource Manager



ARM Template

```
"parameters": {
  "<parameter-name>" : {
    "type" : "<type-of-parameter-value>",
    "defaultValue": "<default-value-of-parameter>",
    "allowedValues": [ "<array-of-allowed-values>" ],
    "minValue": <minimum-value-for-int>,
    "maxValue": <maximum-value-for-int>,
    "minLength": <minimum-length-for-string-or-array>,
    "maxLength": <maximum-length-for-string-or-array-parameters>,
    "metadata": {
      "description"
    }
  }
}
```

```
"variables": {
  "<variable-name>": "<variable-value>",
  "<variable-name>": {
    <variable-complex-type-value>
  },
  "<variable-object-name>": {
    "copy": [
      {
        "name": "<name-of-array-property>",
        "count": <number-of-iterations>,
        "input": <object-or-value-to-repeat>
```

```
[

me": "<variable-array-name>",
unt": <number-of-iterations>,
put": <object-or-value-to-repeat>
```

```
"functions": [
  {
    "namespace": "<namespace-for-functions>",
    "members": {
      "<function-name>": {
        "parameters": [
          {
            "name": "<parameter-name>",
            "type": "<type-of-parameter-value>"
          }
        ],
        "output": {
          "type": "<type-of-output-value>",
          "value": "<function-return-value>"
        }
      }
    }
  }
],
```

```
"resources": [
  {
    "condition": "<true-to-deploy-this-resource>",
    "type": "<resource-provider-namespace/resource-type-name>",
    "apiVersion": "<api-version-of-resource>",
    "name": "<name-of-the-resource>",
    "comments": "<your-reference-notes>",
    "location": "<location-of-resource>",
    "dependsOn": [
      "<array-of-related-resource-names>"
    ],
    "tags": {
      "<tag-name1>": "<tag-value1>",
      "<tag-name2>": "<tag-value2>"
    },
    "sku": {
      "name": "<sku-name>",
      "tier": "<sku-tier>",
      "size": "<sku-size>",
      "family": "<sku-family>",
      "capacity": <sku-capacity>
    },
    "kind": "<type-of-resource>",
    "copy": {
      "name": "<name-of-copy-loop>",
      "count": <number-of-iterations>,
      "mode": "<serial-or-parallel>",
      "batchSize": <number-to-deploy-serially>
    },
    "plan": {
      "name": "<plan-name>",
      "promotionCode": "<plan-promotion-code>",
      "publisher": "<plan-publisher>",
      "product": "<plan-product>",
```

```
"outputs": {
  "<output-name>": {
    "condition": "<boolean-value-whether-to-output-value>",
    "type": "<type-of-output-value>",
    "value": "<output-value-expression>",
    "copy": {
      "count": <number-of-iterations>,
      "input": <values-for-the-variable>
    }
  }
}
```

ARM Template – Reference Guide

Microsoft | Docs Documentation Learn Q&A Code Samples Search

Azure Product documentation Architecture Learn Azure Develop Resources Portal Free account

Azure / Azure Templates Bookmark Share

Filter by title

- Reference
  - AAD
  - Compute
    - Availability Sets
    - Cloud Services
    - Disk Accesses
    - Disk Encryption Sets
    - Disks
    - Galleries
    - Host Groups
    - Images
    - Proximity Placement Groups
    - Snapshots
    - Ssh Public Keys
    - Virtual Machines

Define resources in ARM templates

12/21/2020 • 2 minutes to read •

When creating Azure Resource Manager temp
understand what resource types are available,
ARM template reference documentation prov

Learn how to create te

For an introduction to working with templates
ARM template .

To learn about ARM templates through a guic
Deploy and manage resources in Azure by usi

Microsoft recommends that you use VS Code
Azure Resource Managed tools extension, you
For more information, see Quickstart: Create A

Is this page helpful?

Microsoft.Network virtualNetworks

12/28/2020 • 12 minutes to read •

API Versions: Latest

Is this page helpful? Yes No

In this article
- Template format
- Property values
- Quickstart templates

Template format

To create a Microsoft.Network/virtualNetworks resource, add the following JSON to the resources section of your template.

JSON

```
{
  "name": "string",
  "type": "Microsoft.Network/virtualNetworks",
  "apiVersion": "2020-07-01",
  "location": "string",
  "tags": {},
  "extendedLocation": {
    "name": "string",
    "type": "EdgeZone"
  },
  "properties": {
    "addressSpace": {
      "addressPrefixes": [
        "string"
      ]
    },
```

VirtualNetworkPropertiesFormat object

Name	Type	Required	Value
addressSpace	object	No	The AddressSpace that contains an array of IP address ranges that can be used by subnets. - AddressSpace object
dhcpOptions	object	No	The dhcpOptions that contains an array of DNS servers available to VMs deployed in the virtual network. - DhcpOptions object
subnets	array	No	A list of subnets in a Virtual Network. - Subnet object
ipAllocations	array	No	Array of IpAllocation which reference this VNET. - SubResource object

Is this page helpful? Yes No

In this article
- Template format
- Property values
- Quickstart templates

ARM Template learning path

https://bit.ly/3qZGNj1

MODULE 1: Deploy Azure infrastructure by using ARM templates

MODULE 2: Deploy to multiple Azure environments by using ARM template features

MODULE 3: Preview changes and validate Azure resources by using what-if and the ARM template test toolkit

MODULE 4: Automate the deployment of ARM templates by using GitHub Action

MODULE 5: Extend ARM templates by using deployment scripts

MODULE 6: Manage complex cloud deployments by using advanced ARM template features

ARM Template – Azure Quickstart Templates

Microsoft Azure

Contact Sales | Search | My account | Portal | Esther

Overview | Solutions | Products | Documentation | Pricing | Training | Marketplace | Partners | Support | Blog | More | Free account

Azure Quickstart Templates

Deploy Azure resources through the Azure Resource Manager with community contributed templates to get more done. Deploy, learn, fork and contribute back.

Sort by:

- Date updated
- Template name
- Author name
- Most popular

Resource Types:

- All
- Microsoft.Aad (1)
- Microsoft.Analysisservices (1)
- Microsoft.Apimanagement (11)
- Microsoft.Appconfiguration (2)
- Microsoft.Appplatform (1)
- Microsoft.Attestation (1)
- Microsoft.Authorization (15)

Search

Showing all 978 templates. Refine

Deploy a simple Windows VM

This template allows you to deploy a simple Windows VM using a few different options for the Windows version, using the latest patched version. This will deploy an A2 siz...

by Brian Moore,

Last updated: 9/18/2020

Create an new AD Domain with 2 Domain Controllers

This template creates 2 new VMs to be AD DCs (primary and backup) for a new Forest and Domain

Templates / Deploy a simple Windows VM

Deploy a simple Windows VM

by Brian Moore

Last updated: 9/18/2020

Deploy to Azure | Browse on GitHub

This template allows you to deploy a simple Windows VM using a few different options for the Windows version, using the latest patched version. This will deploy an A2 size VM in the resource group location and return the FQDN of the VM.

Parameters

PARAMETER NAME	DESCRIPTION
OSVersion	The Windows version for the VM. This will pick a fully patched image of this given Windows version.
vmSize	Size of the virtual machine.
location	Location for all resources.
vmName	Name of the virtual machine.

Use the template

PowerShell

```
New-AzResourceGroup -Name <resource-group-name> -Location <resource-group-location> #use this command when you need to create a new resource group for your deployment
New-AzResourceGroupDeployment -ResourceGroupName <resource-group-name> -TemplateUri https://raw.githubusercontent.com/Azure/azure-quickstart-templates/master/101-vm-simple-windows/azuredeploy.
```

Install and configure Azure PowerShell

ARM Template – Automatic deployments

```
# ARM Template file
$jsonARMTemplateFile = ".\ARM-WVDNewHostpool.template.json"
$jsonARMParameterFile = ".\ARM-WVDNewHostpool.parameter.json"

# Create WVD Hostpool, based on ARM Template
New-AzResourceGroupDeployment -ResourceGroupName "rg-wvd-infra" `
    -TemplateFile $jsonARMTemplateFile `
    -TemplateParameterFile $jsonARMParameterFile `
    -administratorAccountPassword $secureDomainAdminPassword `
    -vmAdministratorAccountPassword $secureLocalAdminPassword `
    -Verbose
```

ARM Template – Automatic deployments

Deployment failed. Click here for details →

Your deployment failed

Deployment name: ARM-WVDN
Subscription:
Resource group: rg-wvd-infra

Deployment details (Downloa

Resource
vmCreation-linkedTemplate-
AVSet-linkedTemplate-
Workspace-linkedTemplate-
wvd-hp-demo-DAG
wvd-hp-demo

Deployment failed. Click here for details →

Your deployment failed

Deployment name: vmCreation-linkedTemplate-
Subscription:
Resource group: rg-wvd-resources

Start time: 1/27/2021, 5:37:38 PM
Correlation ID: d26c3483-452c-462e-8838-8ba39d8490be

Deployment details (Download)

Resource	Type	Status	Operation details
wvd-sh-0-nic	Microsoft.Network/networkI...	BadRequest	Operation details
wvd-sh-1-nic	Microsoft.Network/networkI...	BadRequest	Operation details
NSG-linkedTemplate	Microsoft.Resources/deploy...	OK	Operation details

ARM Template – Automatic deployments

```
{
  "code": "DeploymentFailed",
  "message": "At least one resource deployment operation failed. Please list deployment operations for details. Please see https://aka.ms/DeployOperations for usage details.",
  "details": [
    {
      "code": "InvalidResourceReference",
      "message": "Resource /subscriptions/ /resourceGroups/rg-wvd-resources/providers/Microsoft.Network/virtualNetworks/vnet-wvd-resource/subnets/default referenced by resource /subscriptions/ /resourceGroups/rg-wvd-resources/providers/Microsoft.Network/networkInterfaces/wvd-sh-0-nic was not found. Please make sure that the referenced resource exists, and that both resources are in the same region."
    },
```

ARM Template – Automatic deployments

```
1   {
2       "$schema": "https://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json#",
3       "contentVersion": "1.0.0.0",
        Select or create a p
4   >   "parameters":
324     },
325 >   "variables":
346     },
347     "resources":
348         {
349             "apiV
350             "name
351             "type
352             "resou
353             "cond
354 >           "prope
369             }
370         },
371 >       { ...
403         },
404         {
405             "apiV
406             "name
407             "resou
408             "deper
409                 "
410             ],
411             "type
412 >           "prope
522             }
523         }
524     ],
525     "outputs": {
526         "rdshVmNa
527             "valu
528             "type
529         }
530     }
531 }
```

```
{
    "apiVersion": "2018-05-01",
    "name": "[concat('vmCreation-linkedTemplate-', parameters('deploymentId'))]",
    "resourceGroup": "[parameters('vmResourceGroup')]",
    "dependsOn": [
        "[concat('AVSet-linkedTemplate-', parameters('deploymentId'))]"
    ],
    "type": "Microsoft.Resources/deployments",
    "properties": {
        "mode": "Incremental",
        Linked template
        "templateLink": {
            "uri": "[variables('vmTemplateUri')]",
            "contentVersion": "1.0.0.0"
        },
        "parameters": {
            "artifactsLocation": {
                "value": "[parameters('artifactsLocation')]"
            },
            "vmImageVhdUri": {
                "value": "[parameters('vmImageVhdUri')]"
            },
            "storageAccountResourceGroupName": {
                "value": "[parameters('storageAccountResourceGroupName')]"
            },
            "vmGalleryImageOffer": {
                "value": "[parameters('vmGalleryImageOffer')]"
            },
            "vmGalleryImagePublisher": {
                "value": "[parameters('vmGalleryImagePublisher')]"
            },
            "vmGalleryImageSKU": {
                "value": "[parameters('vmGalleryImageSKU')]"
            }
        }
    }
}
```

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

ARM Template – Automatic deployments

Your deployment is complete

Deployment name: AddVMsToHostPool-a4695839-60ac-4048-8c0...
Subscription: Microsoft Azure Sponsorship
Resource group: rg-wvd-infra

Start time: 1/30/2021, 4:39:42 PM
Correlation ID: 24959bf7-49ef-4a12-8abf-f42d14bc57b3

Deployment details (Download)

Resource	Type
vmCreation-linkedTemplate-a4	Microsoft.Resources/deploy..
AVSet-linkedTemplate-a46958	Microsoft.Resources/deploy..

Next steps

Go to resource

Your deployment is complete

Deployment name: vmCreation-linkedTemplate-a4695839-60ac-40...
Subscription: Microsoft Azure Sponsorship
Resource group: rg-wvd-resources

Start time: 1/30/2021, 4:39:52 PM
Correlation ID: 24959bf7-49ef-4a12-8abf-f42d14bc57b3

Deployment details (Download)

Resource	Type	Status	Operation details
wvd-sh-1/dscextension	Microsoft.Compute/virtualM...	OK	Operation details
wvd-sh-0/dscextension	Microsoft.Compute/virtualM...	OK	Operation details
wvd-sh-0/joindomain	Microsoft.Compute/virtualM...	OK	Operation details
wvd-sh-1/joindomain	Microsoft.Compute/virtualM...	OK	Operation details
wvd-sh-1	Microsoft.Compute/virtualM...	OK	Operation details
wvd-sh-0	Microsoft.Compute/virtualM...	OK	Operation details
wvd-sh-0-nic	Microsoft.Network/networkI...	Created	Operation details
wvd-sh-1-nic	Microsoft.Network/networkI...	Created	Operation details
NSG-linkedTemplate	Microsoft.Resources/deploy...	OK	Operation details

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'

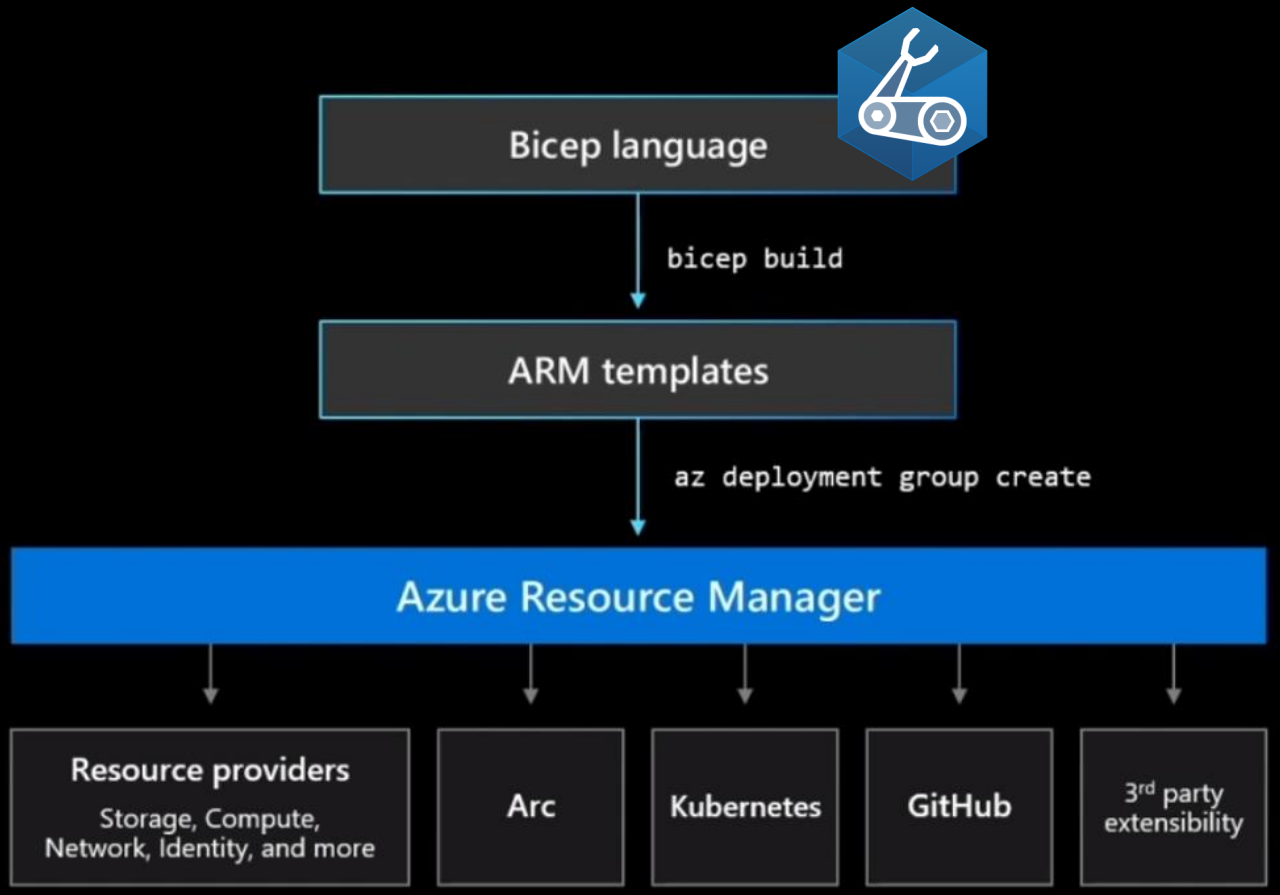
Simple declarative language to provision infrastructure to Azure.

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

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.

Demo – Putting Bicep into (GitHub) Action

cognitionIT/AzureWVD: Collectic | Resource groups - Microsoft Azu

https://github.com/cognitionIT/AzureWVD

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

cognitionIT / AzureWVD

Unwatch 3 Star 8 Fork 0

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

master 2 branches 0 tags Go to file Add file Code

cognitionIT latest version 354b5b9 23 minutes ago 80 commits

.github/workflows	latest version	23 minutes ago
AIBCustomizations	added vscode	19 days ago
ARMTemplates	Adding ARMTemplates	4 months ago
AzModule	Demo Ready!	4 months ago
Bicep	testing bicep	7 days ago
InlineScripts	latest version	23 minutes ago
RESTAPI	RESTAPI Update	6 months ago
.gitignore	Demo ready!!!	4 months ago
LICENSE	Initial commit	7 months ago
README.md	Update README.md	4 months ago

About

Collection of scripts, templates and other resources to support automation of the Windows Virtual Desktop (WVD) Service provided by Microsoft Azure

Readme

MIT License

Releases

No releases published

Create a new release

Packages

No packages published

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 ARM deployment with bicep #15

Summary

Jobs

ARM Deployment with bicep file

ARM Deployment with bicep file

succeeded 21 days ago in 6m 54s

Search logs

- 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  ***
97  Version             Name                                Repository           Description
98  -------             ----                                ----------           -----------
99  5.6.0               Az                                  PSGallery            Microsoft Azure PowerShell - Cmdlets to …
100
101 Script execution Complete
```

- Deploy ARM Resources with Bicep & Azure PowerShell 23s
- Post Checkout 3s
- Complete job 0s

Road map

Current release: CLI version 0.3.167

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
- Linter (TTK successor)
- Snippets & resource scaffolding
- Merging ARM Quickstarts & bicep
- IncludeFile() support

v0.5++

- Module Registry

And then…

Announcing the preview of PSArm

Steve

March 31st, 2021

Announcing PSArm preview

Today, we are pleased to announce the first preview of a new experimental module that make it easier than ever for PowerShell customers to create Azure Resource Manager (ARM) templates: PSArm.

This module enables users to author ARM templates using PowerShell. Similar to Azure Bicep, PSArm is an independent module that creates the necessary ARM JSON template to deploy and configure Azure infrastructure in a PowerShell context. PSArm allows PowerShell users who are familiar with ARM to write complex deployment templates by mixing the declarative syntax of ARM with the iterative syntax of PowerShell.

PSArm

wvd-backplane.psarm.ps1

PSArm > wvd-backplane.psarm.ps1

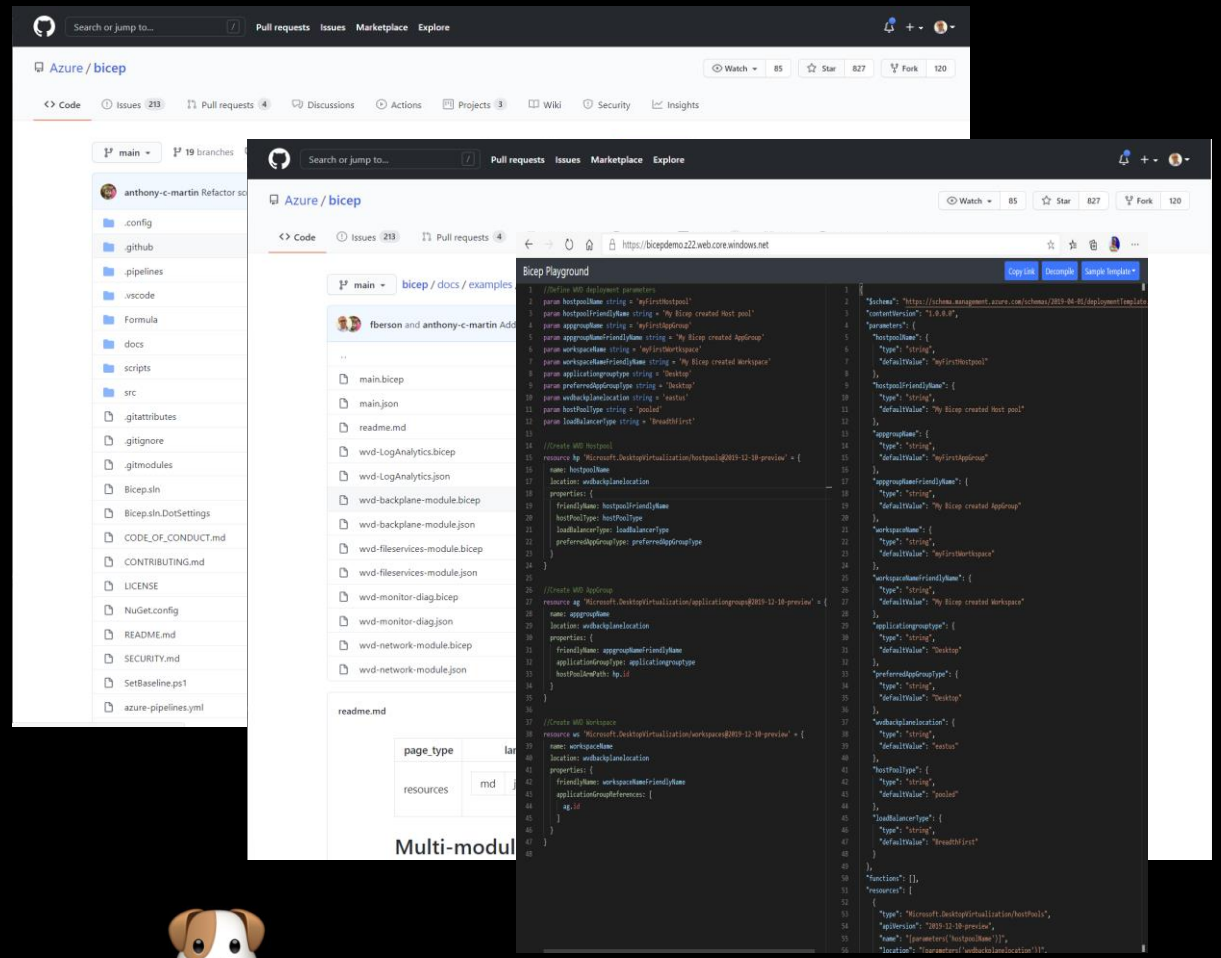
```
Arm {
  param(

    [ArmParameter[string]]
    $hostpoolFriendlyName = 'My Bicep created Host pool',

    [ArmParameter[string]]
    $appgroupNameFriendlyName = 'My Bicep created AppGroup',

    [ArmParameter[string]]
    $workspaceNameFriendlyName = 'My Bicep created Workspace',

    [ValidateSet('Desktop', 'RemoteApp')]
    [ArmParameter[string]]
    $applicationgrouptype = 'Desktop',

    [ValidateSet('Desktop', 'RailApplications')]
    [ArmParameter[string]]
    $preferredAppGroupType = 'Desktop',

    [ArmParameter[string]]
    $wvdbackplanelocation = 'eastus',

    [ArmParameter[string]]
    $hostPoolType = 'pooled',

    [ArmParameter[string]]
    $loadBalancerType = 'BreadthFirst'
  )

  Resource $hostpoolName -Namespace 'Microsoft.DesktopVirtualization' -Type 'hostPools' -ApiVersion '2019-12-10-preview' -Location $
    properties {
      friendlyName $hostpoolFriendlyName
      hostPoolType $hostPoolType
      loadBalancerType $loadBalancerType
      preferredAppGroupType $preferredAppGroupType
    }
  }
```

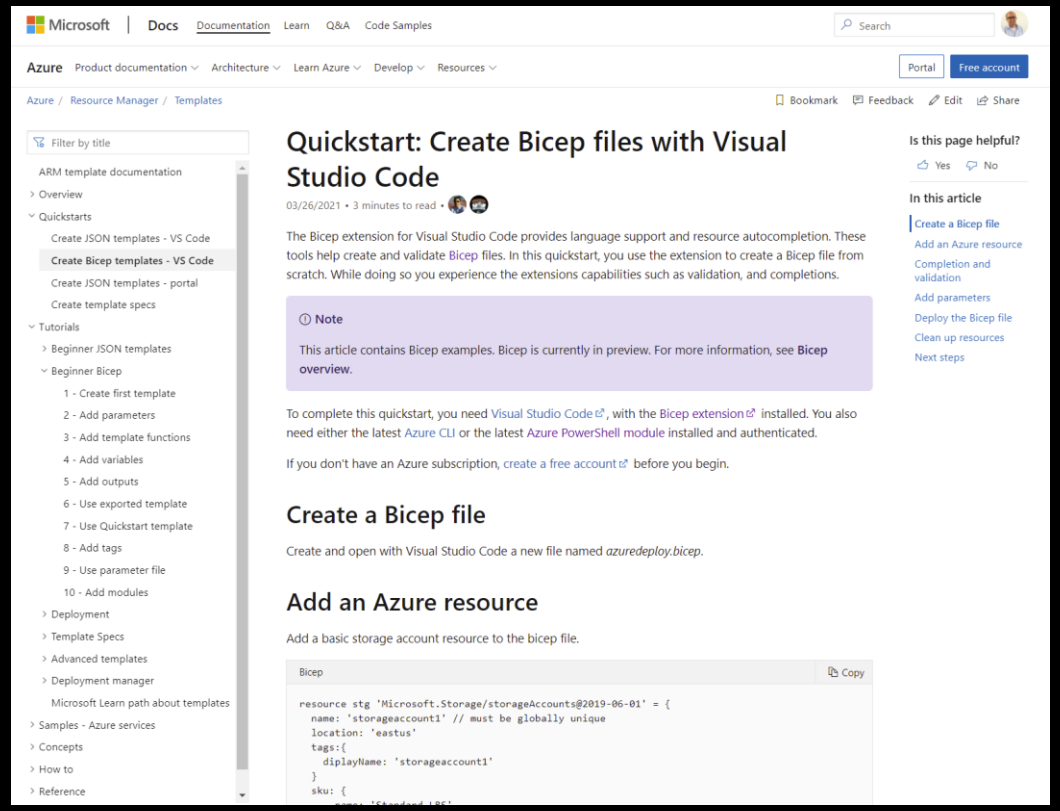
Call to actions:

Install guides, tutorials, example code & playgrounds!

aka.ms/bicep

https://bit.ly/3ml2FnJ





The 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