Authoring ARM Templates

Automated Deployments with ARM

- Deploy, Manage and Monitor all resources in a solution as a group
- Repeatedly deploy a solution throughout the development cycle
- Use declarative templates or imperative scripts
- Define dependencies between resources so they are deployed in the correct order
- Role based access control with all resources
- Use tags to provide further taxonomy of resource groups

JavaScript Object Notation (JSON)

JavaScript Object Notation (JSON)

- Lightweight data-interexchange format based on a subset of JavaScript

- JSON is based on two structure types
  - A collection of name/value pairs.
  - An ordered list of values.

- Supports schemas that enable intellisense/autocomplete in JSON supported editors

JSON Basics

Each JSON file has a start and end bracket

```
{

}
```

JSON objects are defined within the bracket. Syntax for a property is name : value

String values use double quotes "

```
{
  "name": "value",
  "inchesInFoot": 12
}
```

JSON Types

A number (integer or floating point)

A string (in double quotes)

A Boolean (true or false)

An array (in square brackets)

An object (in curly braces)

null

Defining Arrays

Arrays are a single property with multiple values
Values are defined within [ ] brackets

```
{
"availableColors": [
    "blue",
    "red",
    "white"
  ]
}
```

Arrays of Objects

Object properties are nested within { } which are then nested within [ ] brackets

```
"people": [
   {
     "firstName": "bob",
     "favoriteColor": "red"
   },
   {
     "firstName": "fred",
     "favoriteColor": "blue"
   },
   {
     "firstName": "jane",
     "favoriteColor": "green"
   }
 ]
```

Condensed Syntax

Newlines are not required

So be prepared for condensed examples

```
"people": [
   { "firstName": "bob", "favoriteColor": "red" }, { "firstName": "fred", "favoriteColor": "blue" }
 ]
```

Anatomy of an ARM Template

- $schema
  - The URL to the JSON schema that defines the version of the template language
- contentVersion
  - Version of your template. This is useful to ensure you are deploying the correct version of the template
- parameters
  - Define inputs for the template
- variables
  - Custom values usually created from parameters or output from other templates
- resources
  - What resources in Azure (VMs, Databases, etc) the template actually defines
- outputs
  - Return values (if any) that the template produces

```
{
    "$schema":
"https://schema.management.azure.com/schem
as/2015-01-01/deploymentTemplate.json#",

    "contentVersion": "1.0.0.0",

    "parameters": {
    },

    "variables": {
    },

    "resources": [
    ],

    "outputs": {
    }
```

Template Parameters

- Input options that can be specified at template execution time.
- Can be overridden with separate parameters files

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

Parameter Types

- Allowed Types
  - string or secureString - any valid JSON string
  - int - any valid JSON integer
  - bool - any valid JSON boolean
  - object or secureObject - any valid JSON object
  - array - any valid JSON array

Parameters Examples

```
"StorageAccountUniqueName": {
   "type": "string",
   "metadata": {
     "description": "Unique name of storage account"
   }
 },
```

Parameter description

```
"storageAccountType": {
           "type": "string",
           "defaultValue": "Standard_LRS",
           "allowedValues": [
               "Standard_LRS",
                "Standard_GRS",
               "Standard_RAGRS",
               "Premium_LRS"
           ]
       }
```

Default value and allowedValues

```
"instanceCount": {
     "type": "int",
     "minValue": 2,
     "maxValue": 100,
     "metadata": {
       "description": "Number of VM instances"
     }
   },
```

Minimum and Maximum values

Defining Variables

Named values that can store manipulated values from parameters or other resources

```
"parameters": {
   "username": {
     "type": "string"
   },
   "password": {
     "type": "securestring"
   }
 },
 "variables": {
   "connectionString": "[concat('Name=', parameters('username'),
';Password=', parameters('password'))]"
}
```

In this example **connectionString** is the variable and it is created by concatenating text and the user name and password parameters. It can be referenced in other resources

Helper Functions

Used to manipulate or return data from resources, parameter input or data from other resources

- Arithmetic, Array, Azure specific, Conversion, String, Template helpers

```
 "variables": {
      "usernameAndPassword": "[concat('parameters('username'),parameters('password'))]",
      "authorizationHeader": "[concat('Basic ', base64(variables('usernameAndPassword')))]"
    }
```

```
"websiteUri": {
      "type": "string",
      "value": "[concat('http://',reference(resourceId('Microsoft.Web/sites',
parameters('siteName'))).hostNames[0])]"
    }
```

```
"VMStorageName": "[concat('VMStorage', uniqueString(resourceGroup().id))]",
```

```
"location": "[resourceGroup().location]",
```

Resources

- The actual resource(s) the template will instantiate.
- Resources are defined out of resource providers
- Resources have properties that can be read and set
- Resources can have dependencies on other resources

Microsoft.Compute/virtualMachines

Microsoft.Storage/storageAccounts

Microsoft.Network/networkInterfaces

Microsoft.Network/publicIPAddresses

Microsoft.Network/virtualNetworks

Creating multiple instances of a resource

- copy
  - Defines the number of iterations to make
- copyindex()
  - Returns the current index of the iteration. Used to make unique resource names.

```
"name": "[variables('uniqueStringArray')[copyIndex()]]",
"apiVersion": "2015-05-01-preview",
"copy": {
   "name": "storageLoop",
   "count": 5
},
```

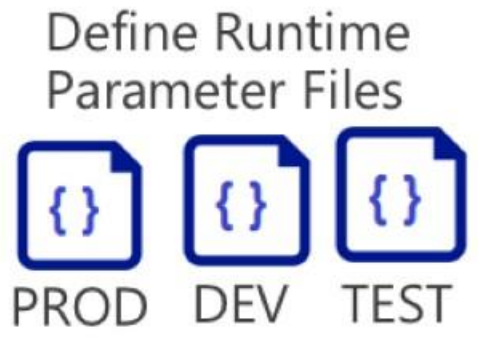
Dependencies

- Use dependencies to control when resources are provisioned.
- For example… a VM could depend on:
  - Storage Account
  - Network Interface
  - Availability Set
  - Script extension for another virtual machine

```
 "dependsOn": [
       "[concat('Microsoft.Storage/storageAccounts/',
parameters('StorageAccountUniqueName'))]",
        "[concat('Microsoft.Network/networkInterfaces/', variables('nicNamePrefix'),
copyindex())]",
        "[concat('Microsoft.Compute/availabilitySets/',
variables('availabilitySetName'))]"
      ],
```

Parameter Files

- Used to store settings specific to an environment



Resource Extensions

- Used to automate the deployed infrastructure
  - Virtual Machine Extensions
    - Custom Script (Windows/Linux)
    - PowerShell DSC (Windows/Linux)
    - Chef, Puppet, AntiMalware, etc….
  - MSDeploy (web deploy)
    - Deploy web deploy package to an Azure Web App
  - SQL Database
    - Deploy a database (.bacpac) to a SQL Database instance

Creating an ARM Template

DEMO

Microsoft Azure

Using Visual Studio to Create and Deploy a Template

Specifying the location of resources

Hard coding the region (not recommended)

```
{
       "location": "West US",
},
```

Using the location of the resource group on deployment

```
{
       "location": "[resourceGroup().location]",
},
```

From a passed in parameter

```
{
       "location": "[parameters('location')]",
},
```

Storage Accounts

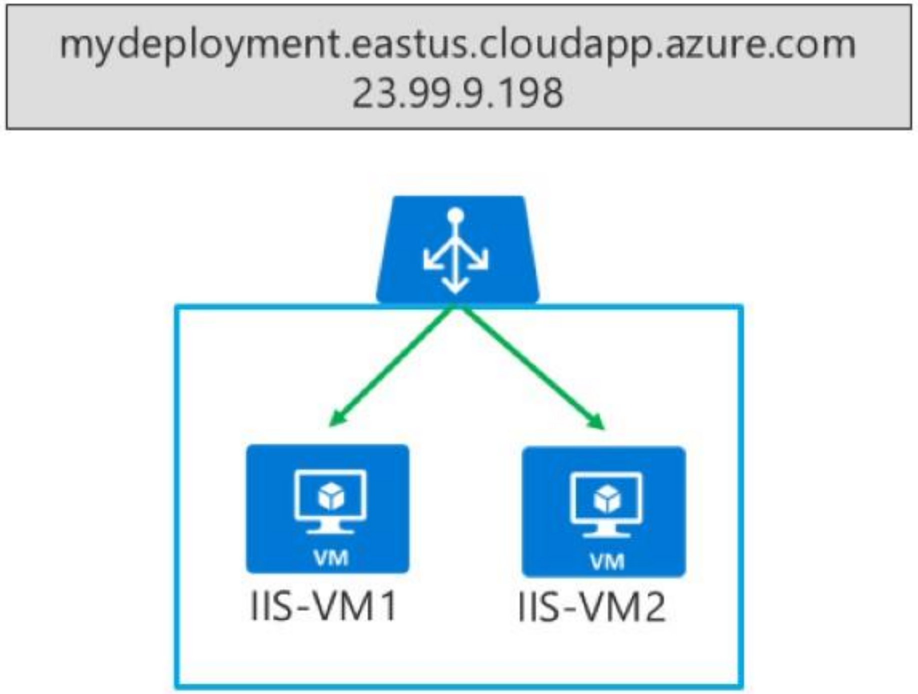
```
{
  "type": "Microsoft.Storage/storageAccounts",
  "name": "variables('StorageAccountName')",
  "apiVersion": "2015-05-01-preview",
  "location": "[resourceGroup().location]",
  "properties": {
     "accountType": "[parameters('storageAccountType')]"
  }
},
```

```
"storageAcountType": {
           "type": "string",
           "defaultValue": "Standard_LRS",
           "allowedValues": [
               "Standard_LRS",
              "Standard_GRS",
               "Standard_RAGRS",
               "Premium_LRS"
           ]
},
```

- Standard_LRS – locally redundant storage
- Standard_GRS – geo redundant storage
- Standard_RGAGRS – read access geo redundant storage
- Premium_LRS – premium local redundant storage

Public IP – Resource Manager

- Optional
- Associate directly with a VM or a Load Balancer
  - Supports up to 100 VMs per load balancer
  - Allows inbound traffic through NAT rules on the load balancer
  - Assign DNS name label (optional)
- Allocation Method
  - Dynamic
  - Static (reserved)
    - Can only be set to static when assigned to a load balancer
    - Guaranteed to remain the same when transferred



Defining a Public IP Address

```
"name": "[variables('PublicIPName')]",
"type": "Microsoft.Network/publicIPAddresses",
"location": "[resourceGroup().location]",
"apiVersion": "2015-05-01-preview",
"dependsOn": [ ],
"tags": {
    "displayName": "PublicIP"
},
"properties": {
```

Can be Static is associated with a LB

```
    "publicIPAllocationMethod": "Dynamic",
```

Up to 30 minutes

```
    "idleTimeoutInMinutes": 4,

    "dnsSettings": {
```

Unique DNS name for the IP (optional)

```
      "domainNameLabel": "[parameters('PublicIPDnsName')]",
```

FQDN that resolves to the IP and registered in DNS as a PTR record (optional)

```
      "reverseFqdn": "opsgility.com"
    }
}
```

Virtual Network Basics

Address space(s) for the virtual network

Subnet definition(s)

```
"name": "OpsTrainingVNET",
"type": "Microsoft.Network/virtualNetworks",
"location": "[resourceGroup().location]",
"apiVersion": "2015-05-01-preview",
"dependsOn": [],
"properties": {
    "addressSpace": {
        "addressPrefixes": [
            "10.0.1.0/24"
        ]
    },
    "subnets": [
        {
            "name": "AppSubnet",
            "properties": {
              "addressPrefix": "10.0.1.0/27"
            }
        }
    ]
```

Specifying DNS

Specifying DNS servers as an array

```
"properties": {
       "addressSpace": {
         "addressPrefixes": [
           "10.0.1.0/24"
         ]
       },
       "dhcpOptions": {
         "dnsServers": [
           "10.0.0.4",
           "8.8.8.8"
         ]
       },
       "subnets": [
         {
           "name": "AppSubnet",
           "properties": {
             "addressPrefix": "10.0.1.0/27"
           }
         }
```

Virtual Machine Resource Provider

Resource type, name and location

```
"type": "Microsoft.Compute/virtualMachines",
"name": "[parameters('VMName')]",
"location": "[resourceGroup().location]",
 "properties": {
```

Size of the virtual machine

```
"hardwareProfile": {
  "vmSize": “[parameters('VMSize')]"
},
```

The availability set name

```
 "availabilitySet": {
    "id": "[parameters('AvailabilitySetName')]“
 },
```

Computer name and credentials

```
 "osProfile": {
      "computername": "[parameters('VMName')]",
      "adminUsername": "[parameters('adminUsername')]",
      "adminPassword": "[parameters('adminPassword')]"
 },
```

Specify the OS and Data Disks

```
'storageProfile": {
  "imageReference": {
    "publisher": "[variables('ImagePublisher')]",
    "offer": "[variables('ImageOffer')]",
    "sku": "[variables('ImageSKU')]",
    "version": "latest"
  },
  "osDisk": {
    "name": "osdisk",
    "vhd": {
      "uri": "[concat('http://',parameters('storageAccount'),'.blob.core.windows.net/'disks/',parameters('VMName'),'-osdisk.vhd')]"
     },
    "caching": "ReadWrite",
    "createOption": "FromImage"
  },
  "dataDisks": [
  {
    "vhd": {
      "uri": "[concat('http://',parameters('storageAccount'),'.blob.core.windows.net/'disks/',parameters('VMName'),'-data1.vhd')]"
     },
     "name":"data-disk1')]",
     "caching" : "None",
     "createOption": "empty",
     "diskSizeGB": 1023,
     "lun": 0
```

Configure an image (or disk)

Configure the OS disk

Configure one or more data disks

Virtual Machine Images

```
"imageReference": {
        "publisher": "MicrosoftSQLServer",
        "offer": "SQL2014-WS2012R2",
        "sku": "Standard",
        "version": "latest"
    },
```

Publisher	Offer	SKU
Microsoft	SQL Server 2008 R2	Enterprise
redhat	SQL Server 2012 SP2	Standard
Barracuda	SQL Server 2012 R2 SP2	Web
Microsoft® SQL Server®	SQL Server 2014	Enterprise Optimized
	SQL Server 2016	Enterprise Optimized DW
		Enterprise Optimized OLTP

Querying Images

PowerShell

```
Get-AzureRmVMImagePublisher `
        -Location $locName |
        Select PublisherName

Get-AzureRmVMImageOffer -Location $loc `
    -PublisherName $publisher

Get-AzureRmVMImageSku -Location $loc `
    -PublisherName $publisher `
    -Offer $offer
```

CLI 1.0

```
azure vm image list-publishers

azure vm image list-offers

azure vm image list-skus
```

CLI 2.0

```
az vm image list --all
az vm image list --offer Debian -o table --all
```

VM Network Adapters

```
{
  "name": "variables('VMNicName')",
  "type": "Microsoft.Network/networkInterfaces",
  "location": "[resourceGroup().location]",
  "apiVersion": "2015-05-01-preview",
  "dependsOn": [
    "[concat('Microsoft.Network/virtualNetworks/', 'vnetNa[illegible]e')]"
  ],
  "tags": {
    "displayName": "VMNic"
  },
  "properties": {
    "ipConfigurations": [
      {
        "name": "ipconfig1",
        "properties": {
          "privateIPAllocationMethod": "Dynamic",
          "subnet": {
            "id": "[variables('SubnetRef')]"
          }
        }
      }
    ]
  }
},
```

```
"networkProfile": {
    "networkInterfaces": [
    {
        "id":
"[resourceId(resourceGroup().name,'Microsoft.Network/networkInterfa
ces', [variables('VMNicName')]]"
     }
    ]
},
```

Defining Static IPs

privateIPAllocationMethod: Static

privateIPAddress: Static IP address assigned from the subnet

```
"SubnetRef": "[concat(variables('VnetID'), '/subnets/', variables('OPSTrainingVNETAppsName'))]",
```

```
"ipConfigurations": [
      {
        "name": "ipconfig1",
        "properties": {
         "privateIPAllocationMethod": "Static",
         "privateIPAddress" :"[parameters('VMIP')]",
         "subnet": {
          "id": "[variables('SubnetRef')]"
         }
        }
      }
```

10.0.0.100

Apps - 10.0.0.0/24

Resource Extensions

Resource extensions are denoted using the resources property on a virtual machine, or under extensions on VM scale sets.

Accepts a list of resource extensions to apply to the resources.

Examples: Scripts, Chef, Puppet, DSC

```
"resources": [
 {
    "name": "DSCEXT",
    "type": "extensions",
    "location": "[resourceGroup().location]",
    "apiVersion": "2015-05-01-preview",
    "dependsOn": [
       "[concat('Microsoft.Compute/virtualMachines/', parameters('VMName'))]"
    ],
    "tags": {
       "displayName": "DSCEXT"
    },
```