Managing ARM Templates.

- Understanding Azure Resource Manager (ARM)
- Exporting and Importing ARM templates.
- ARM Resource Providers
- Deploy ARM Templates
    - Using PowerShell
    - Azure CLI
    - Azure Portal
    - REST API
- Incremental and Complete Deployments
- Linked templates
- Nested templates

About Azure Resource Manager Templates

About Resource Provider

Each resource provider offers a set of resources and operations for working with an Azure service. For example, if you want to store keys and secrets, you work with the Microsoft.KeyVault resource provider. This resource provider offers a resource type called vaults for creating the key vault.

The name of a resource type is in the format: **{resource-provider}/{resource-type}.** For example, the key vault type is **Microsoft.KeyVault/vaults**.

Before getting started with deploying your resources, you should gain an understanding of the available resource providers. Knowing the names of resource providers and resources helps you define resources you want to deploy to Azure. Also, you need to know the valid locations and API versions for each resource type.

About Azure Resource Manager:

The infrastructure that makes up your application is often composed of various different components. For instance, you might simply be running a web site, but behind the scenes you have an Azure web site deployed, a Storage account for tables, blobs, and queues, a couple of VMs running a database cluster, etc…

The old way of doing things was to use the **Service Management API**. Each piece of infrastructure was **treated separately** in the Azure Portal. Deployment consisted of manually creating them, or lots of effort creating scripts using PowerShell. You would have to handle trying to initialize infrastructure in serial or parallel yourself. If you

Deccansoft Software Services H.No: 153, A/4, Balamrai, Rasoolpura, Secunderabad-500003 TELANGANA, NDIA.
http://www.deccansoft.com | http://www.bestdotnettraining.com
Phone: +91 40 2784 1517 OR +91 8008327000 (INDIA)

wanted to deploy just part of your infrastructure you would have to understand **what was already there** and take actions accordingly, either manually or in your scripts.

Azure Resource Manager Overview:

Azure Resource Manager enables you to work with the resources in your solution as a group. You can deploy, update, or delete all the resources for your solution in a single, coordinated operation.

- **Azure Resource Manager** allows you to define **a declarative template** for your group of resources.
- A resource manager template is **a JavaScript Object Notation (JSON)** file that defines one or more resources to deploy to a resource group.
- You use that template for deployment and it can work for different environments such as **testing, staging, and production** and have confidence your resources are deployed in a consistent state.
- You can define the dependencies between resources so they're deployed in the correct order.
- Azure handles things like **parallelizing** as much of the deployment as possible without any extra effort on your part.
- It also deploys in an idempotent way i.e. **incrementally adds** anything **missing** while leaving anything already deployed in place, so you can **rerun** the template deployment multiple times safely.

Quick Start Templates: https://azure.microsoft.com/en-us/resources/templates/

Azure Resource Manager Templates

Resource Manager enables you to export a Resource Manager template from existing resources in your subscription.

It is important to note that there are **two** different ways to export a template:

1. You can **export the actual template that you used for a deployment**. The exported template includes all the parameters and variables exactly as they appeared in the original template. This approach is helpful when you have deployed resources through the portal. Now, you want to see how to construct the template to create those resources.
    a) Select the Resource Group → **Deployments** in Overview Section → Click on Link provided
    b) Select the deployment and **Template** tab.
2. You can **export a template that represents the current state of the resource group**. It creates a template that is a snapshot of the resource group. The exported template has many hard-coded values and probably not as

Deccansoft Software Services H.No: 153, A/4, Balamrai, Rasoolpura, Secunderabad-500003 TELANGANA, NDIA.
http://www.deccansoft.com | http://www.bestdotnettraining.com
Phone: +91 40 2784 1517 OR +91 8008327000 (INDIA)

many parameters as you would typically define. This approach is useful when you have modified the resource group through the portal or scripts. Now, you need to capture the resource group as a template.

a) Resource Group Properties → select **Export Templates** (Automation)

b) To **edit** the template, you can now either download the template files locally or you can you can save the template to your library and work on it through the portal.

c) Let's continue with later option of saving to library. When adding a template to the library, give the template a name and description. Then, select **Save**

```
{
  "$schema": "https://schema.management.azure.com/schemas/2019-04-01/deploymentTemplate.json#",
  "contentVersion": "",
  "parameters": {  },
  "variables": {  },
  "functions": [  ],
  "resources": [  ],
  "outputs": {  }
}
```

Sample to Create a Storage Account: Template.json

```
{
 "$schema": "https://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json#",
 "contentVersion": "1.0.0.0",
 "parameters": {
  "storageName": {
   "type": "string",
   "minLength": 3,
   "maxLength": 24
  },
  "storageSKU": {
   "type": "string",
   "defaultValue": "Standard_LRS",
   "allowedValues": [
```


Deccansoft Software Services H.No: 153, A/4, Balamrai, Rasoolpura, Secunderabad-500003 TELANGANA, NDIA.
http://www.deccansoft.com | http://www.bestdotnettraining.com
Phone: +91 40 2784 1517 OR +91 8008327000 (INDIA)

```
      "Standard_LRS",
      "Standard_GRS",
      "Standard_RAGRS",
      "Standard_ZRS",
      "Premium_LRS",
      "Premium_ZRS",
      "Standard_GZRS",
      "Standard_RAGZRS"
    ]
  },
  "location": {
    "type": "string",
    "defaultValue": "[resourceGroup().location]"
  }
},
"resources": [
  {
    "type": "Microsoft.Storage/storageAccounts",
    "apiVersion": "2019-04-01",
    "name": "[parameters('storageName')]",
    "location": "[parameters('location')]",
    "sku": {
      "name": "[parameters('storageSKU')]"
    },
    "kind": "StorageV2",
    "properties": {
      "supportsHttpsTrafficOnly": true
    }
  }
],
  "outputs": {
      "storageUri": {
```


Deccansoft Software Services H.No: 153, A/4, Balamrai, Rasoolpura, Secunderabad-500003 TELANGANA, NDIA.
http://www.deccansoft.com | http://www.bestdotnettraining.com

Phone: +91 40 2784 1517 OR +91 8008327000 (INDIA)

```
        "type": "string",
        "value": "[reference(parameters('storageName')).primaryEndPoints.blob]"
      }
    }
}
```

Sample: **Parameters.json**

```
{
 "$schema": "https://schema.management.azure.com/schemas/2015-01-01/deploymentParameters.json#",
 "contentVersion": "1.0.0.0",
 "parameters": {
  "storageName": {
   "value": "devstore"
  },
  "storageSKU": {
   "value": "Standard_LRS"
  }
 }
}
```

Deploy the Template using Portal

To Save the template:

1. More Services → Template Spec → +Add
2. Name="DemoVMTemplate", Description="This is a test template" → OK
3. Copy the content of Storage.json and paste in the Text Editor → OK
4. Click Add.

To Execute the Template

5. More Services → Search for "**Template Spec**"
6. Select the template with the name you saved earlier (DemoVMTemplate)**.**
7. Template Blade → Deploy
8. Click on Edit parameters → Copy and paste content from Storage-Parameters.json → Save

Deccansoft Software Services H.No: 153, A/4, Balamrai, Rasoolpura, Secunderabad-500003 TELANGANA, NDIA.
http://www.deccansoft.com | http://www.bestdotnettraining.com

Phone: +91 40 2784 1517 OR +91 8008327000 (INDIA)

9. Create a New RG and change the parameters as needed → Check I agree . . . → Purchase

Deploy Templates using PowerShell

Azure Portal → Cloud Shell → Upload file

PowerShell commands to Execute the ARM Template

```
$templateFile = "$Home/storage.json"
$parameterFile="$Home/storage-parameters.json"
New-AzResourceGroup `
 -Name myDemoRG `
 -Location "East US"

 New-AzResourceGroupDeployment `
 -Name devenvironment `
 -ResourceGroupName myDemoRG `
 -TemplateFile $templateFile `
 -TemplateParameterFile $parameterFile
```

Note: Validate your deployment settings to find problems before creating actual resources.

Test-AzResourceGroupDeployment -ResourceGroupName ExampleResourceGroup -TemplateFile <PathToTemplate>

You have the following options for providing parameter values:

New-AzResourceGroupDeployment -Name ExampleDeployment -ResourceGroupName ExampleResourceGroup -TemplateFile <PathToTemplate> -TemplateParameterFile $parameterFile **-myParameterName1 "parameterValue1" -myParameterName2 "parameterValue2"**

Use a Parameter object:

$parameters = @{"<ParameterName>"="<Parameter Value>",}

New-AzureRmResourceGroupDeployment -Name ExampleDeployment -ResourceGroupName ExampleResourceGroup -**TemplateFile** <PathToTemplate> -**TemplateParameterObject** $parameters

Deploy the Template using Azure CLI

Deccansoft Software Services H.No: 153, A/4, Balamrai, Rasoolpura, Secunderabad-500003 TELANGANA, NDIA.
http://www.deccansoft.com | http://www.bestdotnettraining.com
Phone: +91 40 2784 1517 OR +91 8008327000 (INDIA)

1. Change to directory where template and parameters file is saved
2. Login to Azure:

   az Login
3. Execute the following to create the resources

```
az group create --name DemoRG --location "South Central US"

az group deployment create \
  --name ExampleDeployment \
  --mode Complete \
  --resource-group DemoRG \
  --template-file template.json \
 --parameter-file parameters.json
```

Note: ***template-file*** parameter can be replaced with ***template-uri*** if the json file is store externally on internet

```
--template-uri "https://www.sites.com/template.json"
```

Incremental and Complete Deployments

When deploying your resources, you specify that the deployment is either an **incremental** update or a **complete** update.

- In complete mode, Resource Manager **deletes** resources that exist in the resource group but are not specified in the template.
- In incremental mode, Resource Manager **leaves unchanged** resources that exist in the resource group but are not specified in the template.

Existing Resource Group contains:

- Resource A
- Resource B
- Resource C

Template defines:

- Resource A
- Resource B
- Resource D

When deployed in **incremental** mode, the resource group contains:

- Resource A
- Resource B
- Resource C
- Resource D

When deployed in **complete** mode, Resource C is deleted. The resource group contains:

- Resource A
- Resource B
- Resource D

PowerShell Command:

```
New-AzureRmResourceGroupDeployment -Name "ExampleDeployment123" -Mode Complete -
ResourceGroupName DemoRG -TemplateFile "D:\template.json" -TemplateParameterFile 'D:\parameters.json'
```

CLI Command:

```
az group deployment create --name ExampleDeployment --mode Complete --resource-group DemoRG --template-
file template.json --parameter-file @parameters.json
```

Create Windows VM using ARM Template

1. Open the following files from DevOps folder
   - AzureVMTemplate.json
   - AzureVMTemplate-parameters.json

There are five resources defined by the template:

1. Microsoft.Storage/storageAccounts.
2. Microsoft.Network/publicIPAddresses.
3. Microsoft.Network/virtualNetworks.
4. Microsoft.Network/networkInterfaces.
5. Microsoft.Compute/virtualMachines.

2. Using Cloud Shell, upload the file AzureVMTempalte.json
3. Copy and Paste the following in Cloud Shell

```
$resourceGroupName = Read-Host -Prompt "Enter the resource group name"
```

```
$storageAccountName = Read-Host -Prompt "Enter the storage account name"
$newOrExisting = Read-Host -Prompt "Create new or use existing (Enter new or existing)"
$location = Read-Host -Prompt "Enter the Azure location (i.e. centralus)"
$vmAdmin = Read-Host -Prompt "Enter the admin username"
$vmPassword = Read-Host -Prompt "Enter the admin password" -AsSecureString
$dnsLabelPrefix = Read-Host -Prompt "Enter the DNS Label prefix"

New-AzResourceGroup -Name $resourceGroupName -Location $location
New-AzResourceGroupDeployment `
  -ResourceGroupName $resourceGroupName `
  -adminUsername $vmAdmin `
  -adminPassword $vmPassword `
  -dnsLabelPrefix $dnsLabelPrefix `
  -storageAccountName $storageAccountName `
  -newOrExisting $newOrExisting `
  -TemplateFile "$HOME/AzureVMTemplate.json"
```

Integrate Key Vault

open *azuredeploy.parameters.json*

```
"adminPassword": {
  "reference": {
    "keyVault": {
    "id":
"/subscriptions/<SubscriptionID>/resourceGroups/<ResGroup>/providers/Microsoft.KeyVault/vaults/<KeyVaultName>"
    },
    "secretName": "vmAdminPassword"
  }
},
```

Create linked Azure Resource Manager templates

- **The main template**: create all the resources except the storage account.

Deccansoft Software Services H.No: 153, A/4, Balamrai, Rasoolpura, Secunderabad-500003 TELANGANA, NDIA.
http://www.deccansoft.com | http://www.bestdotnettraining.com
Phone: +91 40 2784 1517 OR +91 8008327000 (INDIA)

- **The linked template**: create the storage account.

Call the linked template

1. Open AzureVMTemplate.json
2. Replace the storage account resource definition with the following json snippet:

```
{
  "name": "linkedTemplate",
  "type": "Microsoft.Resources/deployments",
  "apiVersion": "2018-05-01",
  "properties": {
    "mode": "Incremental",
    "templateLink": {
      "uri":"https://raw.githubusercontent.com/deccansoft/Demos/master/StorageTemplate.json"
    },
    "parameters": {
      "storageAccountName":{"value": "[variables('storageAccountName')]"},
      "location":{"value": "[parameters('location')]"}
    }
  }
},
```

Pay attention to these details:

- A **Microsoft.Resources/deployments** resource in the main template is used to link to another template.
- The deployments resource has a name called **linkedTemplate**. This name is used for configuring dependency.
- You can only use Incremental deployment mode when calling linked templates.
- templateLink/uri contains the linked template URI. Update the value to the URI you get when you upload the linked template (the one with a SAS token).
- Use parameters to pass values from the main template to the linked template.

3. Make sure you have updated the value of the uri element
4. Expand the virtual machine resource definition, update **dependsOn** and **storageUri** as shown in the following screenshot:

Deccansoft Software Services H.No: 153, A/4, Balamrai, Rasoolpura, Secunderabad-500003 TELANGANA, NDIA.
http://www.deccansoft.com | http://www.bestdotnettraining.com
Phone: +91 40 2784 1517 OR +91 8008327000 (INDIA)

```
"resources": [
  {
    "name": "linkedTemplate",
    "type": "Microsoft.Resources/deployments",
    "apiVersion": "2018-05-01",
    "properties": {…
    }
  },
  {…
  },
  {…
  },
  {…
  },
  {…
  },
  {
    "type": "Microsoft.Compute/virtualMachines",
    "apiVersion": "2018-10-01",
    "name": "[variables('vmName')]",
    "location": "[parameters('location')]",
    "dependsOn": [
      "linkedTemplate",
      "[resourceId('Microsoft.Network/networkInterfaces/', variables('nicName'))]"
    ],
    "properties": {
      "hardwareProfile": {…
      },
      "osProfile": {…
      },
      "storageProfile": {…
      },
      "networkProfile": {…
      },
      "diagnosticsProfile": {
        "bootDiagnostics": {
          "enabled": true,
          "storageUri": "[reference('linkedTemplate').outputs.storageUri.value]"
        }
      }
    }
  }
],
```

5. Azure Portal → Cloud Shell → Upload file
6. PowerShell commands to Execute the ARM Template

```
$templateFile = "$Home/AzureVMTemplateWithLinkToStorage.json"
$parameterFile="$Home/AzureVMTemplateWithLinkToStorage-parameters.json"
New-AzResourceGroup `
 -Name myResourceGroupDev `
 -Location "East US"
New-AzResourceGroupDeployment `
 -Name devenvironment `
 -ResourceGroupName myResourceGroupDev `
 -TemplateFile $templateFile `
 -TemplateParameterFile $parameterFile
```


Deccansoft Software Services H.No: 153, A/4, Balamrai, Rasoolpura, Secunderabad-500003 TELANGANA, NDIA.
http://www.deccansoft.com | http://www.bestdotnettraining.com
Phone: +91 40 2784 1517 OR +91 8008327000 (INDIA)

Creating Resource group and Resources at Subscription level

To create the resource group and deploy resources to it, use a nested template. The nested template defines the resources to deploy to the resource group. Set the nested template as dependent on the resource group to make sure the resource group exists before deploying the resources.

The following example creates a resource group, and deploys a storage account to the resource group.

```
{
 "$schema": "https://schema.management.azure.com/schemas/2018-05-
01/subscriptionDeploymentTemplate.json#",
 "contentVersion": "1.0.0.1",
 "parameters": {
  "rgName": {
   "type": "string",
   "defaultValue": "mydemorg"
  },
  "rgLocation": {
   "type": "string",
   "defaultValue": "eastus"
  },
  "storagePrefix": {
   "type": "string",
   "defaultValue": "dss",
   "maxLength": 11
  }
 },
 "variables": {
  "storageName": "[concat(parameters('storagePrefix'), uniqueString(subscription().id, parameters('rgName')))]"
 },
 "resources": [
  {
   "type": "Microsoft.Resources/resourceGroups",
   "name": "[parameters('rgName')]",
   "apiVersion": "2018-05-01",
```


Deccansoft Software Services H.No: 153, A/4, Balamrai, Rasoolpura, Secunderabad-500003 TELANGANA, NDIA.
http://www.deccansoft.com | http://www.bestdotnettraining.com
Phone: +91 40 2784 1517 OR +91 8008327000 (INDIA)

```
  "location": "[parameters('rgLocation')]",
  "properties": {}
 },
 {
  "name": "storageDeployment",
  "type": "Microsoft.Resources/deployments",
  "apiVersion": "2018-05-01",
  "name": "storageDeployment",
  "resourceGroup": "[parameters('rgName')]",
  "dependsOn": [
   "[resourceId('Microsoft.Resources/resourceGroups/', parameters('rgName'))]"
  ],
  "properties": {
   "mode": "Incremental",
   "template": {
    "$schema": "https://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json#",
    "contentVersion": "1.0.0.0",
    "parameters": {},
    "variables": {},
    "resources": [
     {
      "type": "Microsoft.Storage/storageAccounts",
      "apiVersion": "2017-10-01",
      "name": "[variables('storageName')]",
      "location": "[parameters('rgLocation')]",
      "sku": {
       "name": "Standard_LRS"
      },
      "kind": "StorageV2"
     }
    ],
    "outputs": {}
```

```
    }
   }
  }
 ],
 "outputs": {}
}
```

Execute the Command at the subscription-level

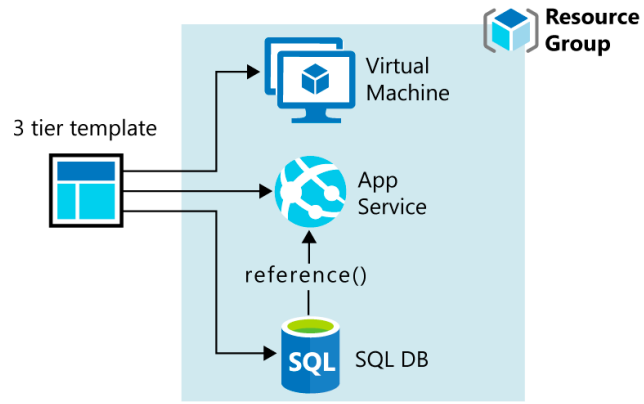
```
New-AzDeployment `
 -Name demoDeployment `
 -Location centralus `
 -TemplateUri $HOME/storagedeploy.json `
```

Note: Parameters have default value.

Note that we have used the command **New-AzDeployment** and not **New-AzResourceGroupDeployment**

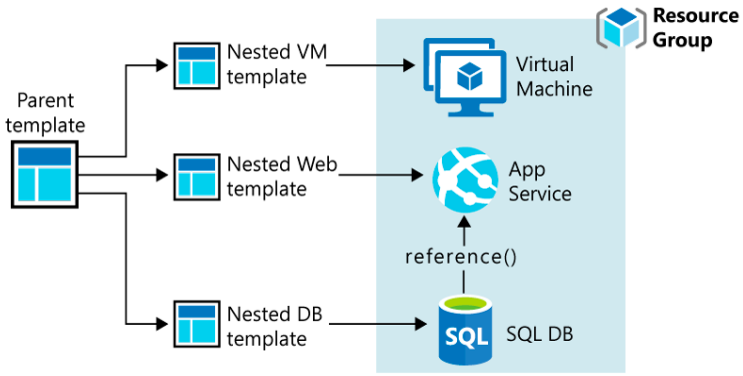
Template Design

You can deploy your three-tier application through a single template to a single resource group.



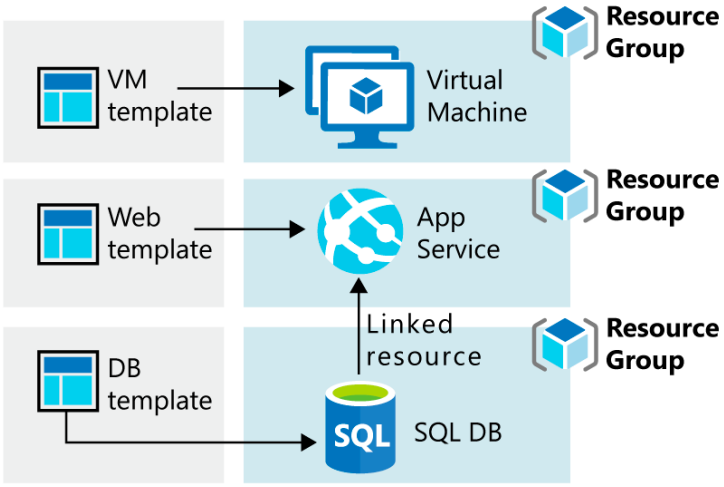
But, you don't have to define your entire infrastructure in a single template. Often, it makes sense to divide your deployment requirements into a set of targeted, purpose-specific templates. You can easily reuse these templates for different solutions. To deploy a solution, you create a master template that links all the required templates. The following image shows how to deploy a three-tier solution through a parent template that includes three nested templates.

Deccansoft Software Services H.No: 153, A/4, Balamrai, Rasoolpura, Secunderabad-500003 TELANGANA, NDIA.
http://www.deccansoft.com | http://www.bestdotnettraining.com
Phone: +91 40 2784 1517 OR +91 8008327000 (INDIA)



If you envision your tiers having separate lifecycles, you can deploy your three tiers to separate resource groups.

Notice the resources can still be linked to resources in other resource groups.



Deccansoft Software Services H.No: 153, A/4, Balamrai, Rasoolpura, Secunderabad-500003 TELANGANA, NDIA.
http://www.deccansoft.com | http://www.bestdotnettraining.com

Phone: +91 40 2784 1517 OR +91 8008327000 (INDIA)