Microsoft

Azure Resource Manager (ARM) Overview

Microsoft Services

Agenda

- ARM & ASM
- Templates
- Resource Groups
- Resource Providers
- Access Control
- Resource Locks
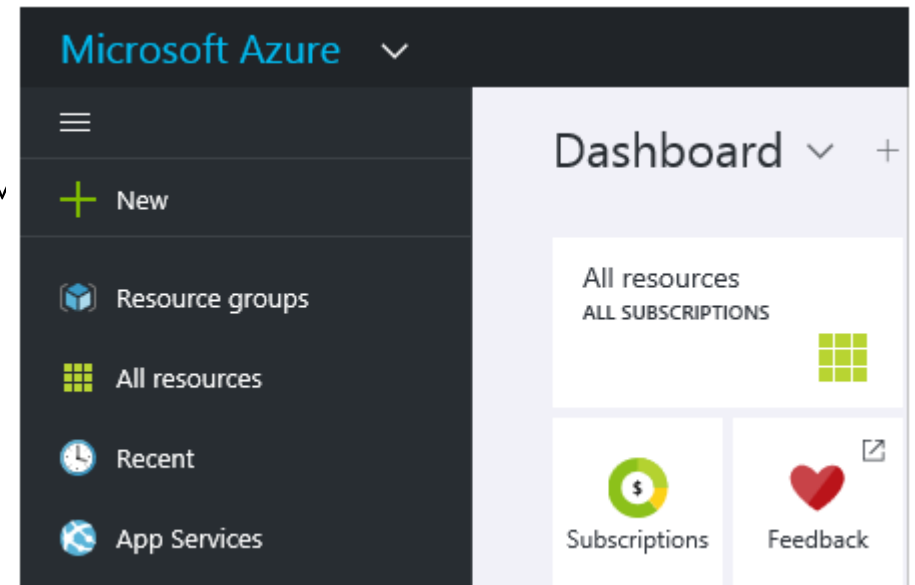- Template Security
- Activity Logs
- Troubleshooting

Azure Service Management (ASM)

- Azure Service Management (ASM) is the initial Azure deployment service that allows you to deploy and manage Azure resources.

- Went GA with Azure in 2010 and did not have a modular approach towards resources.

- Used in the classic & new Azure portals.

- Some of the challenges with ASM are:
  - Serial processing
  - Hard coded resource dependencies
  - No Role Based Access Control (RBAC)
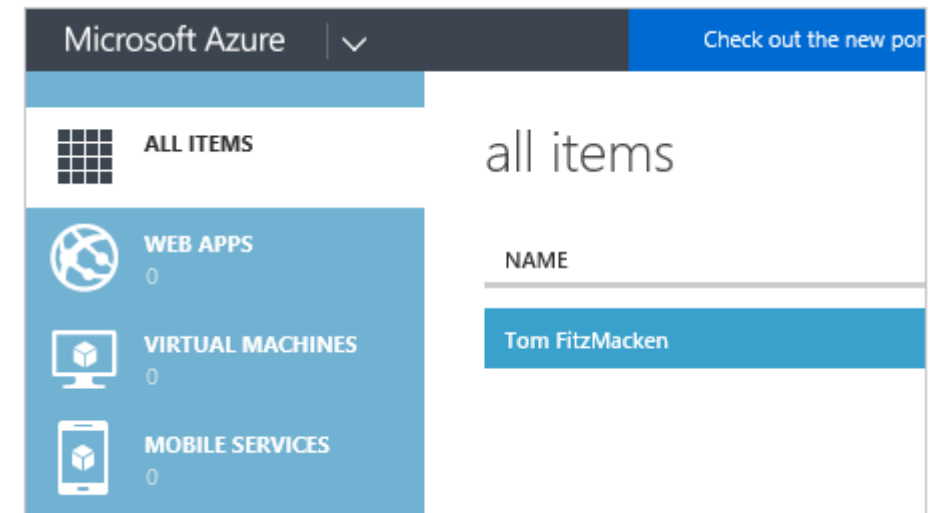
What is Azure Resource Manager (ARM)

- Azure Resource Manager (ARM) is the successor to ASM and allows you to deploy and manage Azure resources in a more improved way.

- Went GA in 2015 and has a modular approach towards resources.

- Used only in the new Azure portal.

- Some of the ASM challenges that are resolved with ARM
  - Parallel processing
  - Deploy resources using templates
  - Resources are not tied to other resources
  - Role Based Access Control (RBAC)

Benefits of Azure Resource Manager

- Deploy, manage, and monitor all of the resources for your solution as a group.

- Redeploy your solution throughout the development lifecycle.

- Manage your infrastructure through templates rather than scripts.

- Administrator defined dependencies.

- Apply access control to all services in your resource group.

Microsoft

Templates

Microsoft Services

Azure Service Management & XML

- Azure Service Management (ASM) is the initial Azure deployment service that allows you to deploy and manage Azure resources.

- ASM uses XML (Extensible Markup Language) to exchange data between a client and the ASM service.

- XML is a data format that is used to exchange data between a web browser and a server.

- XML is verbose, complex and has difficulties exchanging highly structured data.

Azure Resource Manager & JSON

- Azure Resource Manager (ARM) is the successor to ASM and allows you to deploy and manage Azure resources in an improved way.

- ARM uses JSON (JavaScript Object Notation) to exchange data between a client and the ARM service.

- JSON is also a data format that is used to exchange data between a web browser and a server, but it is less verbose, complex and can deal with highly structured data.

XML vs. JSON

- The following are examples that both define an "employees" object, with an array of 3 employee records each:

XML Format

```
<employees>
  <employee>
    <firstName>John</firstName>
<lastName>Doe</lastName>
  </employee>
  <employee>
    <firstName>Anna</firstName>
<lastName>Smith</lastName>
  </employee>
  <employee>
    <firstName>Peter</firstName>
<lastName>Jones</lastName>
  </employee>
</employees>
```

JSON Format

```
{"employees":[
    {"firstName":"John", "lastName":"Doe"},
    {"firstName":"Anna", "lastName":"Smith"},
    {"firstName":"Peter", "lastName":"Jones"}
]}
```

What are ARM Templates

- An ARM template is a file that contains configuration which is passed to Azure Resource Manager for processing.

- The configuration in an ARM template is written in JSON format and saved with a .json extension.

- ARM templates commonly consist of 6 elements; **$schema**, **contentVersion**, **parameters**, **variables**, **resources** and **outputs**.

- The **parameters** element can **optionally** exist in a separate text file called a parameters file which is saved with a .parameters.json extension.

```
MyVnet.json
```

```
Azure Resource Manager
```

```
MyVnet.parameters.json
```
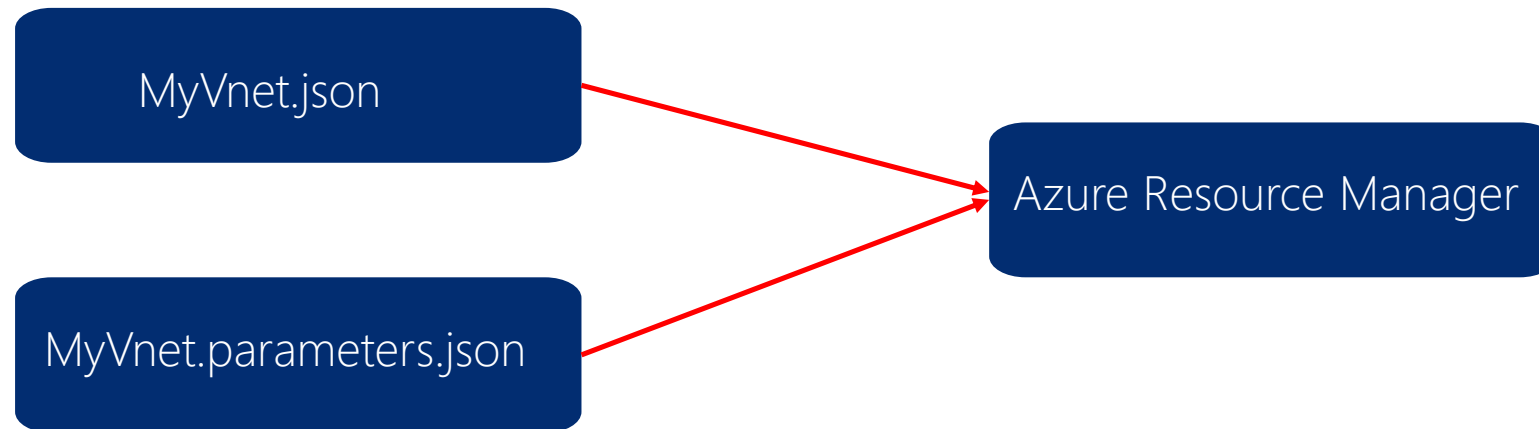
ARM Template Elements

- **$schema** is the location of the JSON schema file that describes the version of the template language.

- **contentVersion** is an arbitrary number that is used to describe the version of the template.

- **Parameters** are values that are provided when deployment is executed in order to customize resource deployment e.g. **"MyStorageAccount"**.

- **Variables** are values that are provided once but are referenced one or more times within an ARM template in order to simplify template language expressions e.g. **"storageAccountName": "StorageAccount1"**.

- **Resources** are used to define the resource types that are deployed or updated in a resource group e.g. a storage account i.e. **Microsoft.Storage/storageAccounts**.

- **Outputs** are values that are returned after deployment.

An ARM Template

- An ARM template in it's simplest structure:

```
{

  "$schema": "http://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json#",

  "contentVersion": "",

  "parameters": {  },

  "variables": {  },

  "resources": [  ],

  "outputs": {  }

}
```

ARM Template Linking

- ARM templates can be linked to other templates to break down the deployment into smaller modules that can be useful for testing and reuse.

- A linked template configuration consists of a **main**, **shared resources** and **member resources** template at a minimum.

- An **optional resources** template and pre-existing **scripts** can also be included.
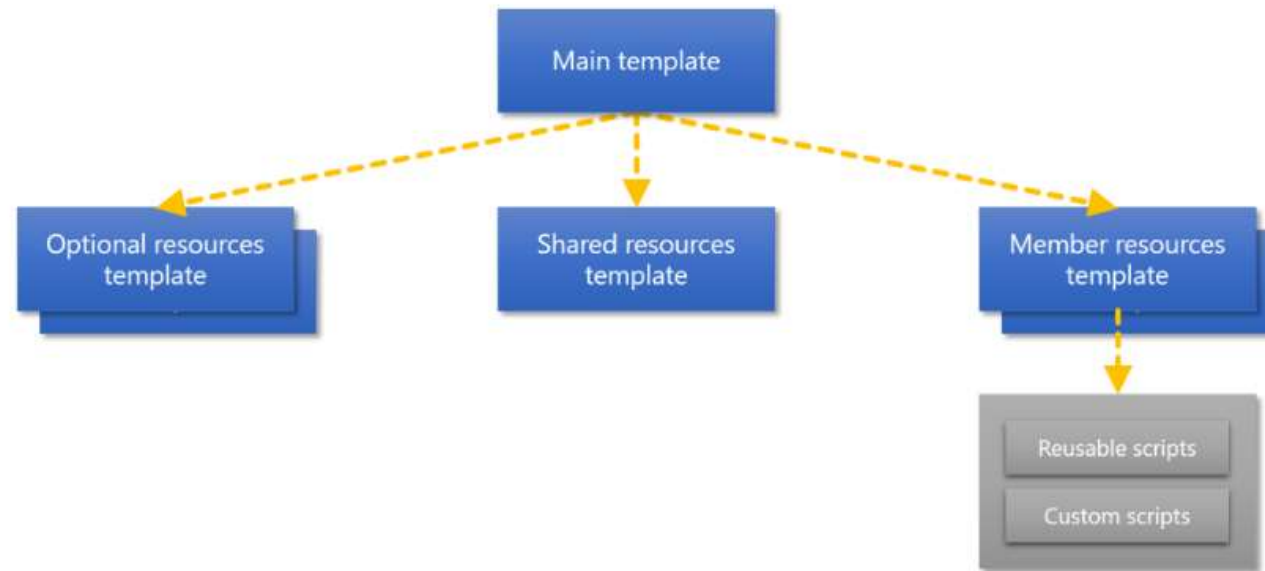
Template Linking Operations & Guidelines

- Parameters are passed from a main template to a linked template for processing.

- The linked template can also pass an output variable back to the source template, enabling a two-way data exchange between templates.

- Use of a local file or a file that is only available on your local network for the linked template is not allowed.

- A URI value that includes either http or https to point to the linked template is allowed.

- Linked templates can be placed in an Azure storage account, and have it's URI used.
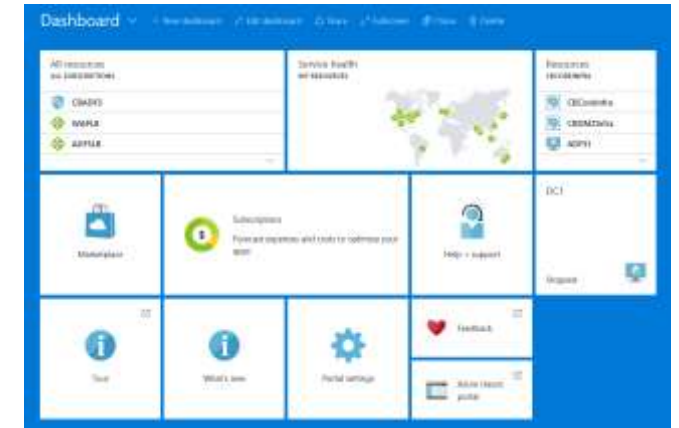
Demo: ARM Templates

Microsoft

Template Authoring

Microsoft Services

Authoring ARM Templates

- ARM templates can be authored using different tools, some of the most common tools in use today are:

  - Visual Studio with Azure SDK
  - New Azure Portal
  - GitHub

- Template files must be <1MB in size.

- Parameter files must be <64 KB in size.

Visual Studio

- One of the most popular tools in use today.

- Provides 14 predefined ARM templates from it's gallery.

- Automatically populates JSON tags when resources are added, but does not remove variables and parameters when resources are removed.
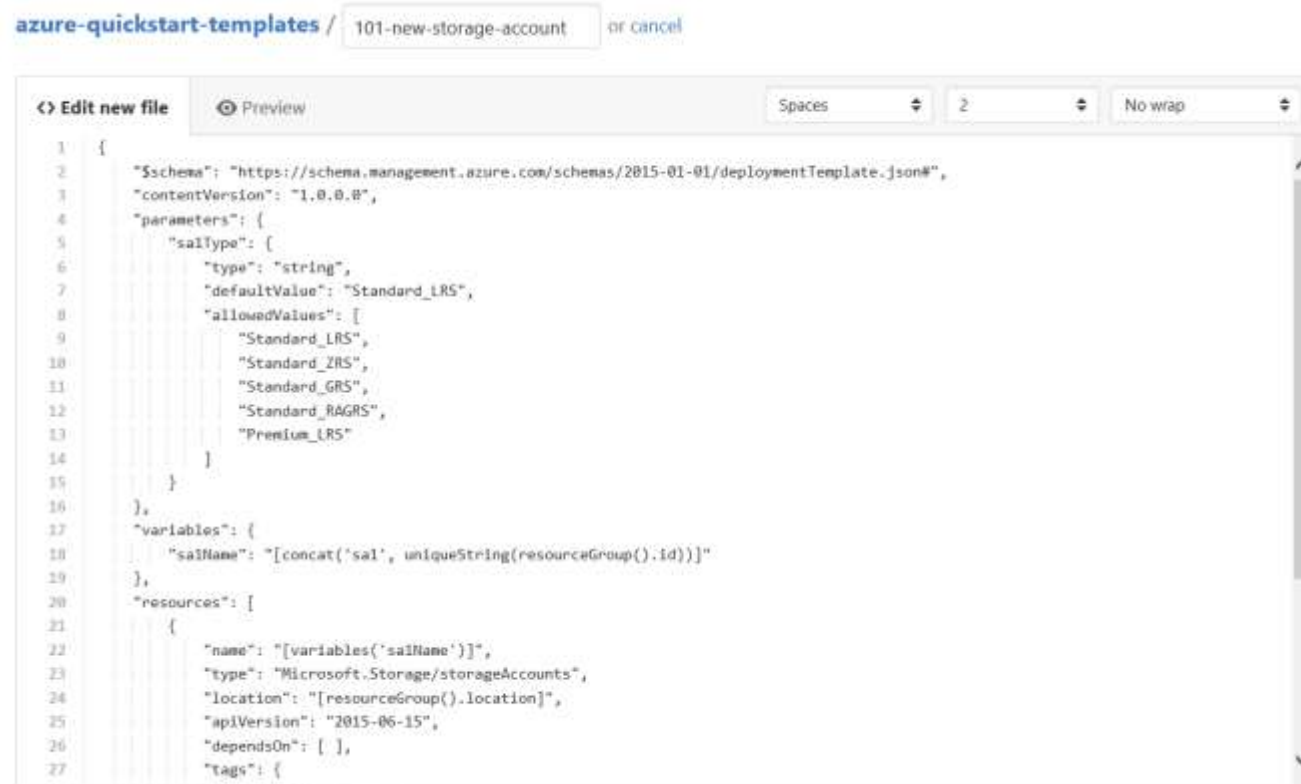
New Azure Portal

- Provides direct access to hundreds of predefined ARM templates in the GitHub gallery that others have created.

- Template parameters can be specified using text boxes.
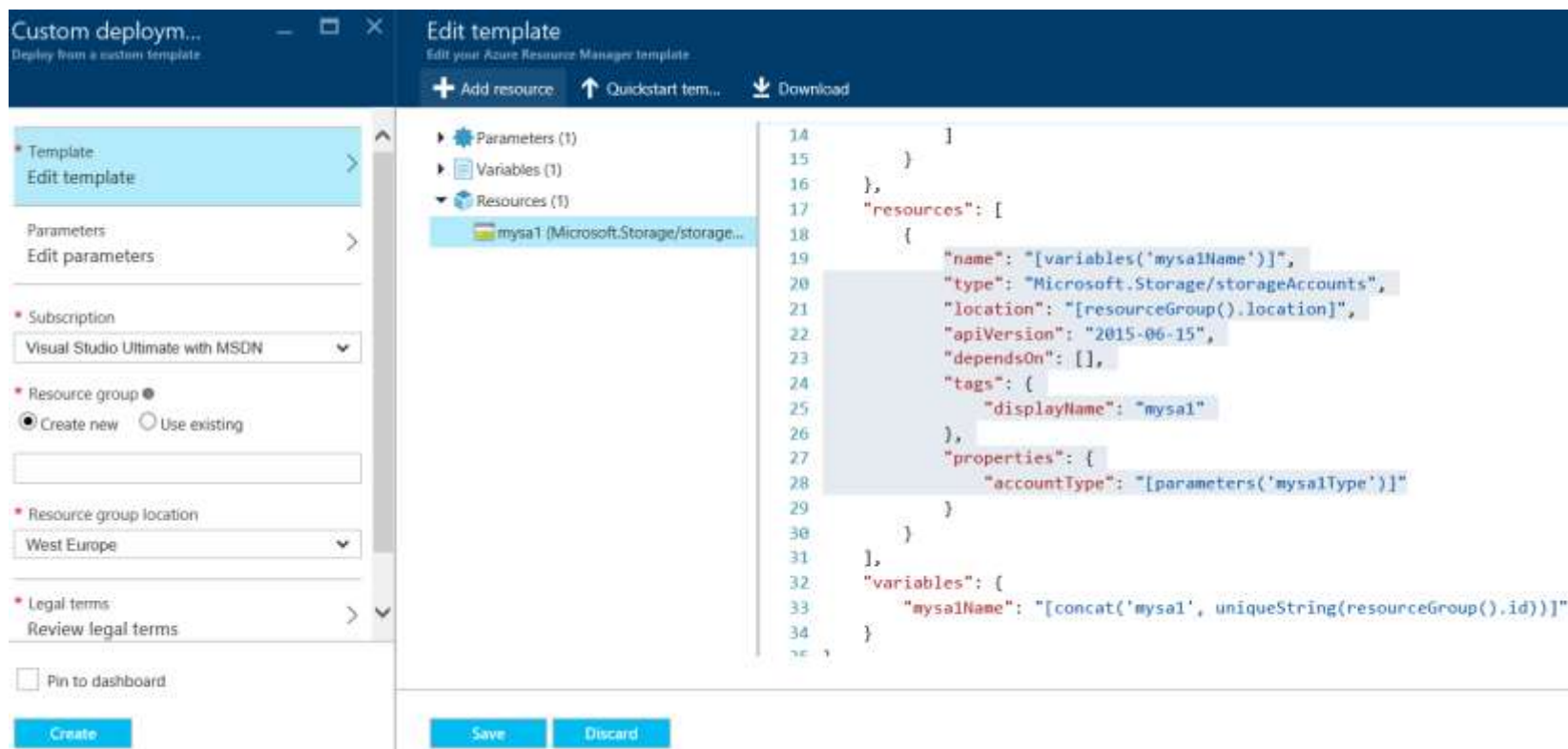
GitHub

- Access to hundreds of predefined ARM templates that others have created.
- No built in syntax error checking or template validation during authoring.

Microsoft

Template Deployment

Microsoft Services

Incremental Deployments

- By default, Resource Manager handles deployments as incremental updates to a resource group.

- Leaves unchanged resources that exist in the resource group but are not specified in the template.

- Adds resources that are specified in the template but do not exist in the resource group.

- Does not re-provision resources that exist in the resource group in the same condition defined in the template.

- Re-provisions existing resources that have updated settings in the template.

Complete Deployments

- Deletes resources that exist in the resource group but are not specified in the template.

- Adds resources that are specified in the template but do not exist in the resource group.

- Does not re-provision resources that exist in the resource group in the same condition defined in the template.

- Re-provisions existing resources that have updated settings in the template.

- Type of deployment specified using the Mode property.

Deploying ARM Templates

- ARM templates can be deployed using different tools, some of the most common tools in use today are:

  - Visual Studio
  - New Azure Portal
  - GitHub
  - PowerShell

- These tools include the deployment of a resource group

  prior to the resource being deployed.

- Templates are validated on deployment.
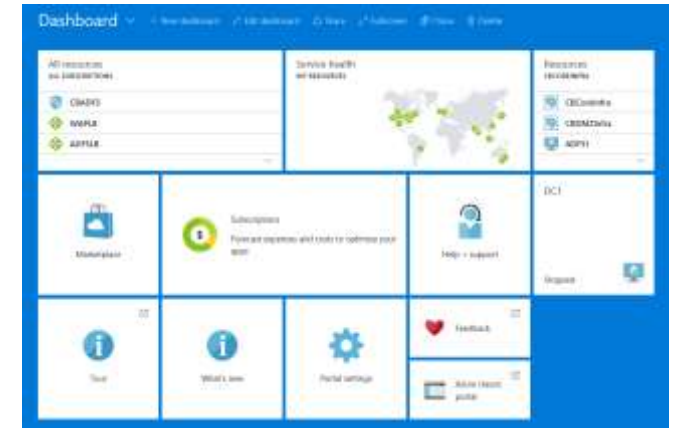
Visual Studio

- Template is deployed using PowerShell from within Visual Studio.
- Deploy using templates from the gallery or redeploy using existing templates.
- Templates are stored locally.

JSON Template

Optional JSON Parameter file

PowerShell script for deployment
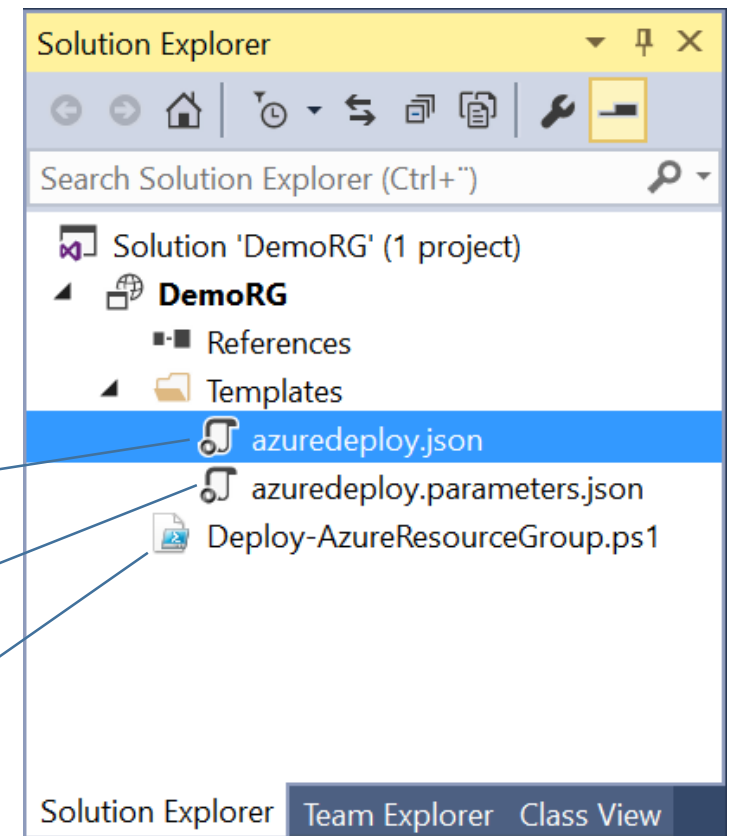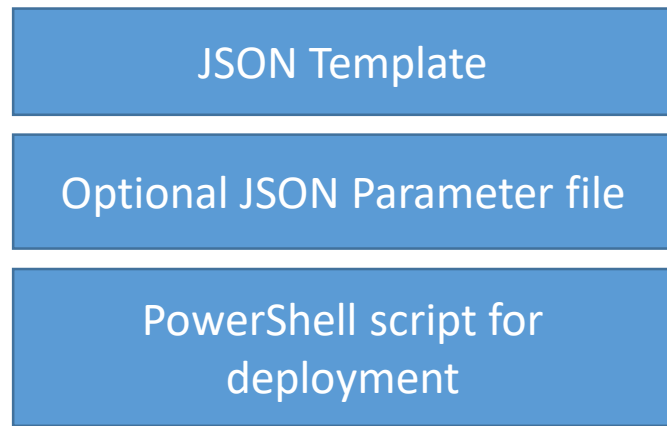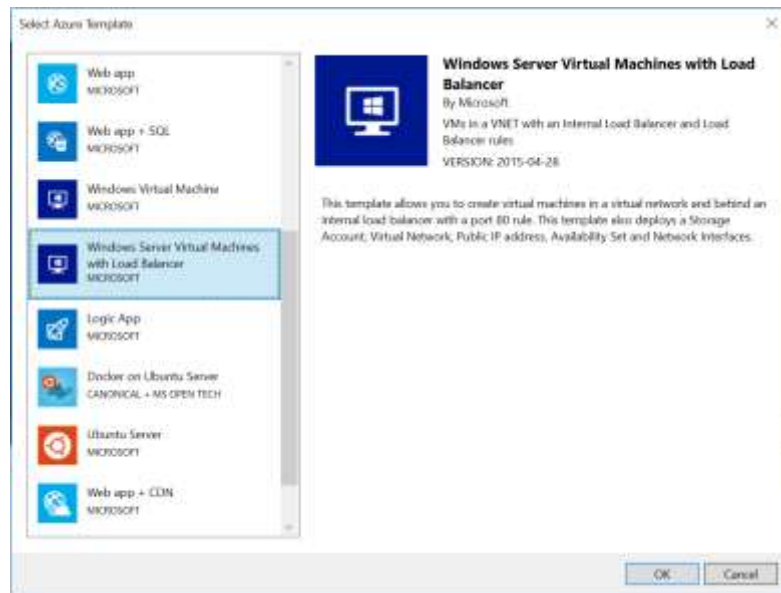
New Azure Portal

- Templates can be deployed, downloaded and reused.

- Templates are stored in Azure.

- Deployed from Marketplace or custom.
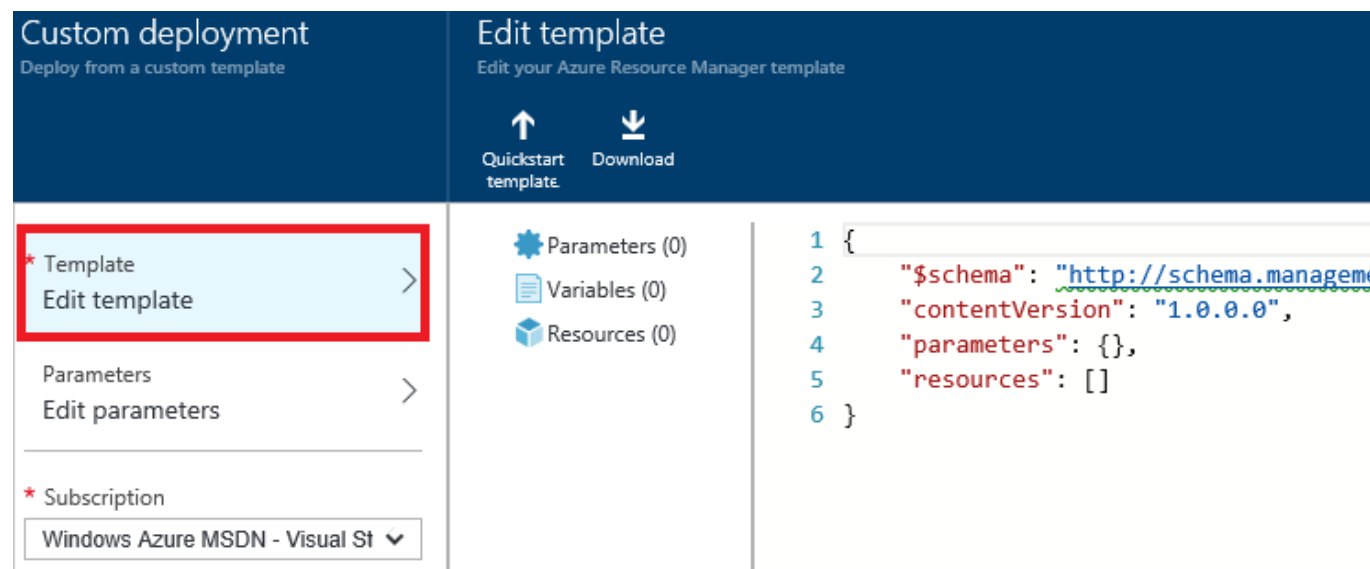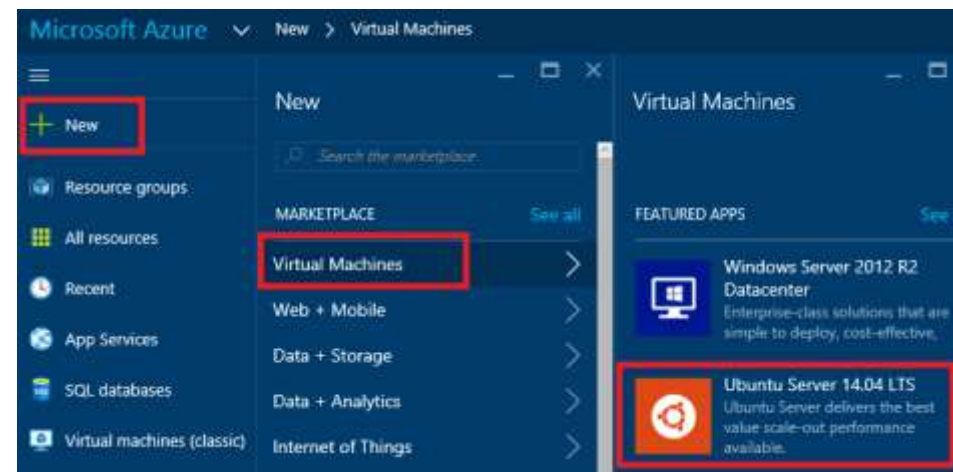
GitHub

- GitHub deploy to Azure button calls new Azure portal and opens a custom deployment blade.

- Templates are stored in GitHub.
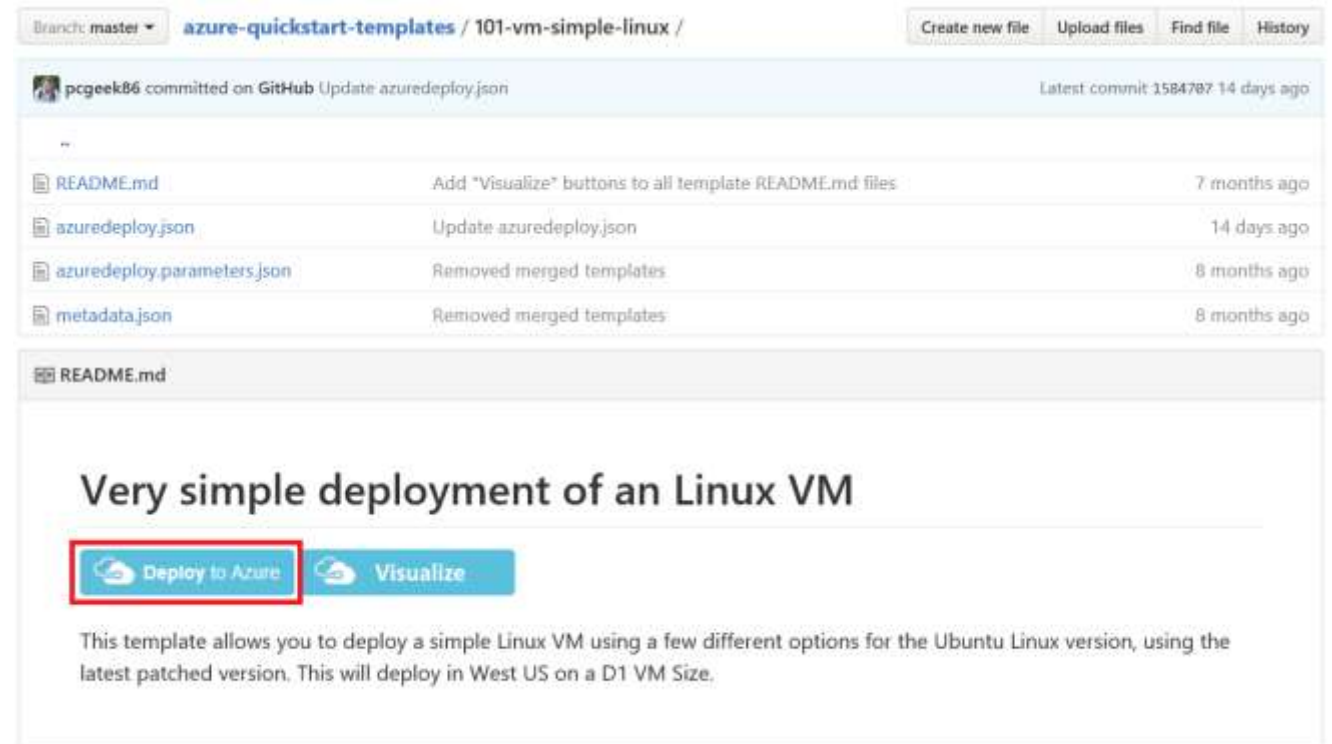
PowerShell With Templates

- New-AzureRmResourceGroup cmdlet creates a new resource group.

- New-AzureRmResourceGroupDeployment cmdlet adds an Azure deployment to an existing resource group.

- Test-AzureResourceGroupTemplate cmdlet verifies a resource group template prior to deployment.

```
PS C:\> New-AzureRmResourceGroupDeployment -Name Deployment1 -ResourceGroupName RG1 -TemplateFile "C:\azuredeploy.json"


DeploymentName          : Deployment1
ResourceGroupName       : RG1
ProvisioningState       : Succeeded
Timestamp               : 24-08-2016 14:38:03
Mode                    : Incremental
TemplateLink            :
Parameters              :

                          Name                  Type                        Value
                          ===============       =========================   ==========
                          mysa1Type             String                      Standard_LRS


Outputs                 :
DeploymentDebugLogLevel :
```

PowerShell Without Templates

- Deploy resources using PowerShell without templates.

- Resource groups must exist prior to resource deployment.

- New-AzureRmVirtualNetwork -ResourceGroupName TestRG -Name TestVNet `

  -AddressPrefix 192.168.0.0/16 -Location westeurope

```
PS C:\> New-AzureRmVirtualNetwork -ResourceGroupName RG1 -Name VNet1 -AddressPrefix 192.168.0.0/16 -Location westeurope

Name                 : VNet1
ResourceGroupName    : RG1
Location             : westeurope
Id                   : /subscriptions/0c224eaf-e4fb-47d9-80ba-d44bacff6ffb/resourceGroups/RG1/providers/Microsoft.Network/virtualN
                       etworks/VNet1
Etag                 : W/"6171c4c5-0de6-47bc-96d8-5039c7212a7e"
ResourceGuid         : d5b4cf84-50b6-4942-b8b4-26dafaf5a5f1
ProvisioningState    : Succeeded
Tags                 :
AddressSpace         : {
                           "AddressPrefixes": [
                             "192.168.0.0/16"
                           ]
                       }
DhcpOptions          : {}
Subnets              : []
VirtualNetworkPeerings : []
```

Demo: Visual Studio Deployment

Microsoft

Resource Groups

Microsoft Services

What are Azure Resource Groups?

- A resource is a manageable item that is available through Azure e.g. a virtual machine, storage account, virtual network and so on.

- A resource group is a container that holds related resources for an application or service, or resources that you group together.

- Each resource group can contain a maximum of 800 resources and can not be nested.

- Each resource can only exist in one resource group.

- Add or remove a resource to a resource group at any time.

- A resource group can contain resources that reside in different regions.

Why Azure Resource Groups?
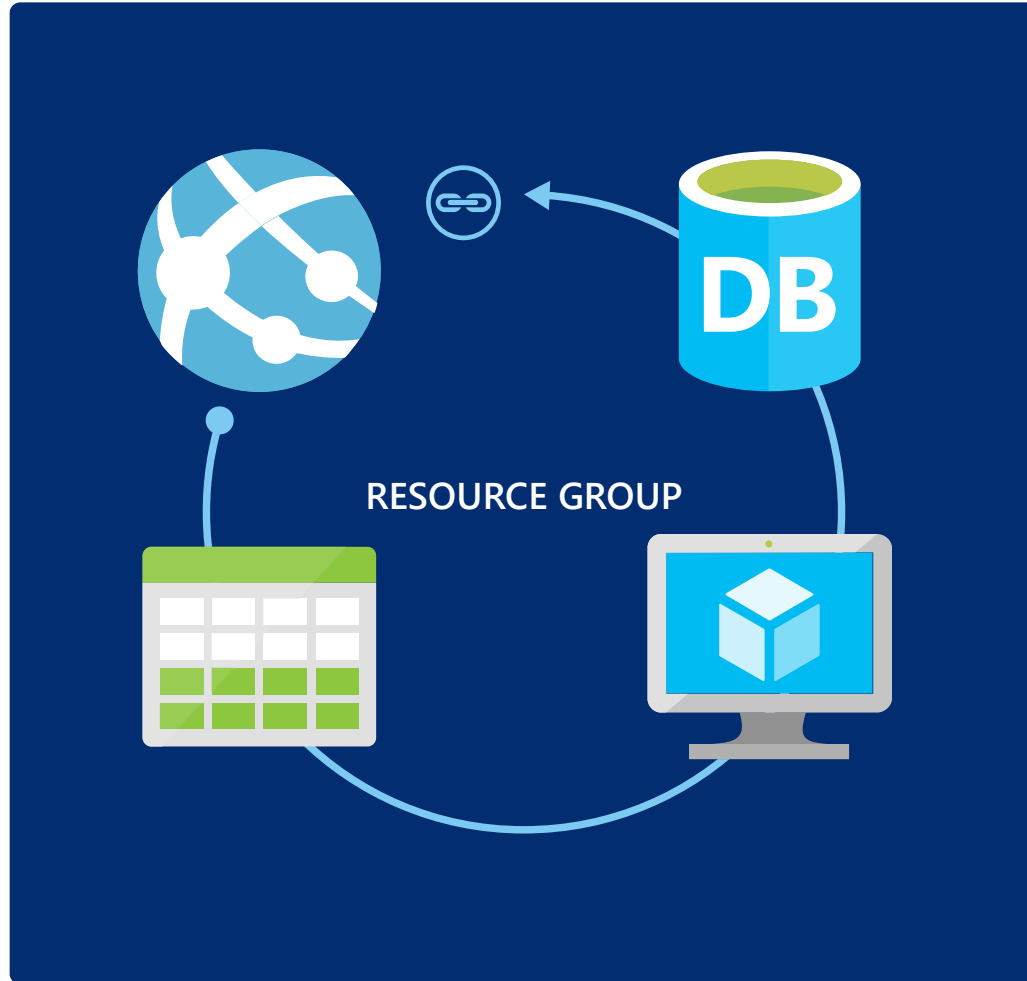
RESOURCE GROUP

- Management – deploy, update, delete and get status on all resources in a resource group from a single point.

- Flexibility - some resources can be moved between resource groups.

- Portability - resource groups can be exported to templates for easier redeployment.

- Billing – a bill can be retrieved on a per resource group basis.

- Access Control – Permissions can be applied on a per resource group basis.

Resource Group Structure

Design:

Should resources be in the same group or a different one?
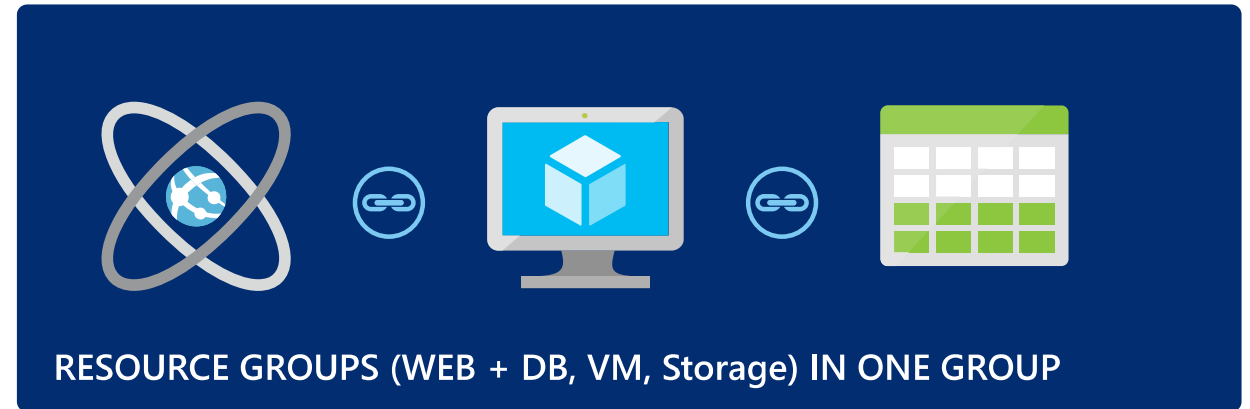
Consideration:

Do they have common lifecycle and management?

RESOURCE GROUPS (WEB + DB, VM, Storage) IN ONE GROUP

OR

WEB + DB RESOURCE GROUP

VIRTUAL MACHINE RESOURCE GROUP

STORAGE RESOURCE GROUP

What are Azure Resource Group Tags?

- A resource group tag consists of a key/value pair that identifies resources with properties that you define e.g. Sales Project: 1

- Each resource group or resource can have a maximum of 15 tags assigned to it.

- The tag name is limited to 512 characters, and the tag value is limited to 256 characters.

- Applied at resource level and are not inherited by child objects i.e. if assigned to a resource group, the resources in that resource group are not tagged.

environment : test

Why Azure Resource Group Tags?
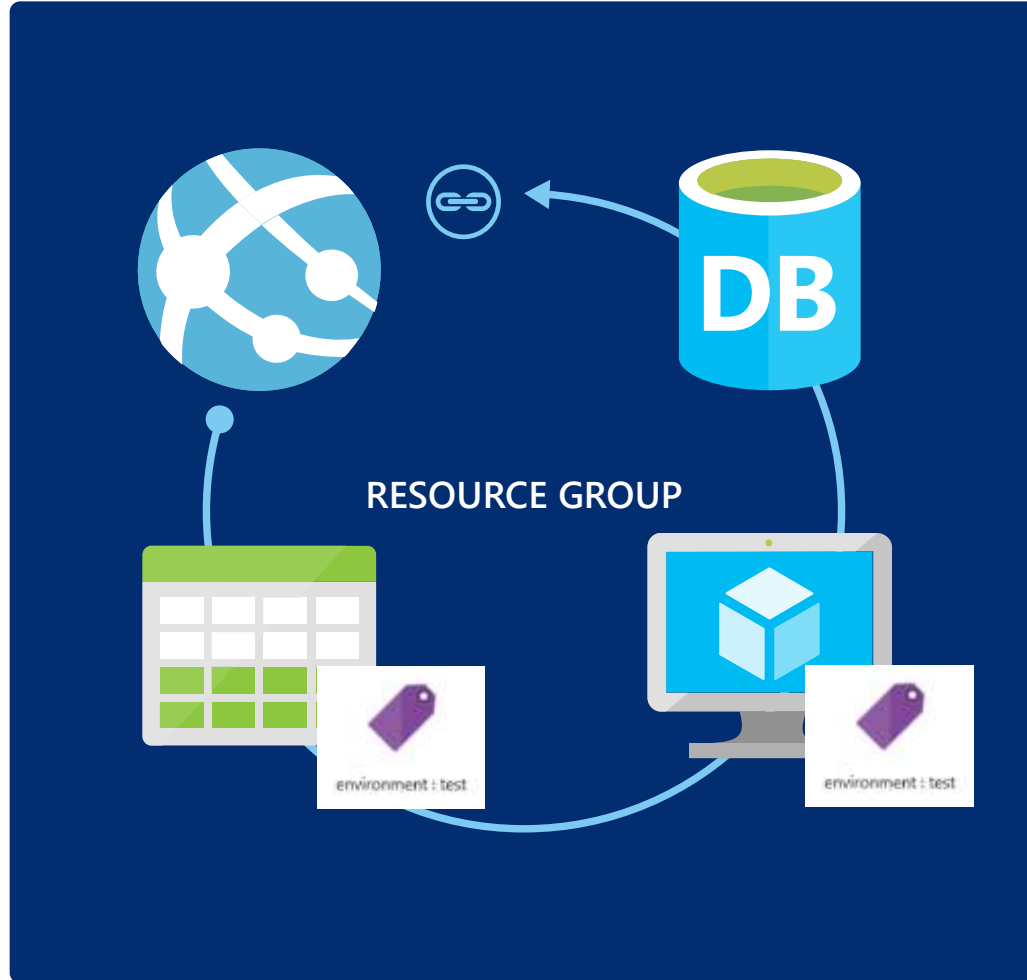
RESOURCE GROUP

- Grouping – resources within or outside a resource group can be further grouped by tagging.

- Billing – for supported services, tags can be used to group billing data e.g. a bill for all resources that are tagged environment:test.

Microsoft

Resource Providers

Microsoft Services

What are Resource Providers?

- A resource provider is a service that supplies the resources that you deploy and manage through Azure Resource Manager.

- Each resource provider offers a set of operations for working with a particular resource type.

- Common resource providers are:
  - Microsoft.Compute which supplies the virtual machine resource.
  - Microsoft.Storage which supplies the storage account resource.
  - Microsoft.Network which supplies the virtual network resource.

- Resource providers have different regional availability and apiVersions.

List available Resource Providers

- To list available resource providers, run:

Get-AzureRmResourceProvider | Format-Table

```
PS C:\> Get-AzureRmResourceProvider | Format-Table

ProviderNamespace               RegistrationState ResourceTypes
-----------------               ----------------- -------------
Microsoft.ApiManagement         Registered        {service, validateServiceName, checkServiceNameAvailability, checkNameA...
Microsoft.Automation            Registered        {automationAccounts, automationAccounts/runbooks, automationAccounts/we...
Microsoft.Backup                Registered        {BackupVault}
Microsoft.Batch                 Registered        {batchAccounts, operations, locations, locations/quotas}
Microsoft.Cache                 Registered        {Redis, locations, locations/operationResults, checkNameAvailability...}
Microsoft.ClassicCompute        Registered        {domainNames, checkDomainNameAvailability, domainNames/slots, domainNam...
Microsoft.ClassicNetwork        Registered        {virtualNetworks, reservedIps, quotas, gatewaySupportedDevices...}
Microsoft.ClassicStorage        Registered        {storageAccounts, quotas, checkStorageAccountAvailability, storageAccou...
Microsoft.Compute               Registered        {availabilitySets, virtualMachines, virtualMachines/extensions, virtual...
microsoft.insights              Registered        {components, webtests, queries, logprofiles...}
Microsoft.KeyVault              Registered        {vaults, vaults/secrets, operations}
Microsoft.MobileEngagement      Registered        {appcollections, appcollections/apps, checkappcollectionnameavailabilit...
Microsoft.Network               Registered        {virtualNetworks, publicIPAddresses, networkInterfaces, loadBalancers...}
Microsoft.OperationalInsights   Registered        {workspaces, workspaces/dataSources, storageInsightConfigs, linkTargets...
Microsoft.SiteRecovery          Registered        {SiteRecoveryVault}
Microsoft.Sql                   Registered        {operations, locations, locations/capabilities, checkNameAvailability...}
Microsoft.Storage               Registered        {storageAccounts, operations, usages, checkNameAvailability...}
Microsoft.VisualStudio          Registered        {account, account/project}
Microsoft.Web                   Registered        {sites/extensions, sites/slots/extensions, sites/instances, sites/slots...
Microsoft.ADHybridHealthService Registered        {services, addsservices, configuration, operations...}
Microsoft.Authorization         Registered        {roleAssignments, roleDefinitions, classicAdministrators, permissions...}
Microsoft.Features              Registered        {features, providers, operations}
Microsoft.Resources             Registered        {tenants, locations, providers, checkresourcename...}
microsoft.support               Registered        {operations, supporttickets}
```

View Resource Providers used by a Subscription

- In the new Azure portal, browse to Resource Explorer then expand Subscriptions/'Your Subscription Name/Providers.
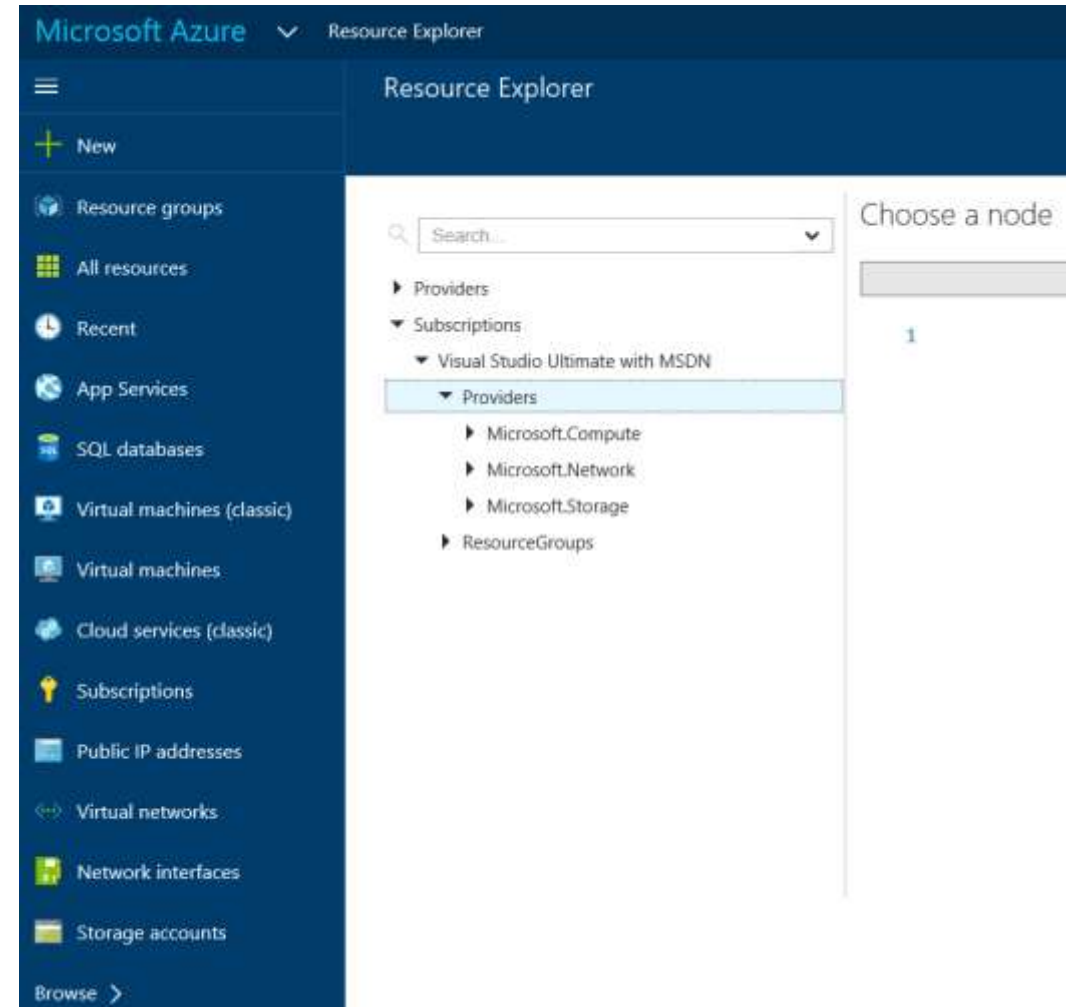
Resource Provider Parameters

- Resource providers require input parameters in order to execute a task.

- Input parameters for the resource being created, read, updated or deleted at a minimum include:

  - Id
    - The Azure wide unique id of the resource. (This includes it's resource group name.)
  - Name
    - The name of the resource e.g. "storageaccount1"
  - Type
    - Describes the type of resource e.g. "virtualNetworks"
  - Location
    - Describes the location of the resource e.g. "westeurope"

Example Resource Provider Parameters

- Example parameters to read, update or delete a virtual network resource VNET1.

    "**id**": "/subscriptions/SubGUID/resourceGroups/RG1/providers/Microsoft.Network/virtualNetworks/VNet1",

    "**name**": "VNet1",

    "**type**": "Microsoft.Network/virtualNetworks",

    "**location**": "westeurope"

Demo: Resource Groups, Resource Tags & View Resource Providers

Management Policies

- ARM Management Policies allow you to enforce polices during resource deployment e.g. specific VM size, Location and Naming Convention.

- MP's compliment RBAC, RBAC is user focused whereas MP's are resource focused.

- Created and managed using PowerShell or REST API.

- Applied at subscription, resource group or resource level and is inherited by all child resources.

- Policy events are audited and can be viewed in the portal or using PowerShell.

- Policies are cumulative.
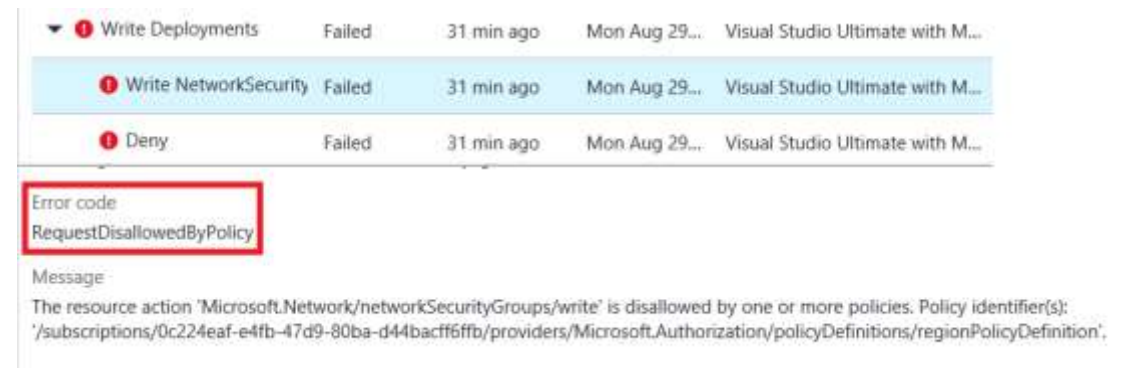
Policy Definition structure

- Policy definition is created using JSON.

- Consists of one or more **conditions/logical operators** which define the actions and an **effect** which tells what happens when the conditions are fulfilled.

- A policy contains the following at a minimum:
  - **Condition/Logical operators:** A set of conditions which can be manipulated through a set of logical operators.
  - **Effect:** This describes what the effect will be when the condition is satisfied – deny, audit or append.

Create a Management Policy

```
$locationpolicy = New-AzureRmPolicyDefinition –Name regionPolicyDefinition -
Description "Policy to allow resource creation in West Europe only" -Policy '{

  "if" : {
    "not" : {
      "field" : "location",
      "in" : ["westeurope"]
    }
  },
  "then" : {
    "effect" : "deny"
  }
}'
```

Assign a Management Policy

New-AzureRmPolicyAssignment -Name locationPolicyAssignment -PolicyDefinition $locationpolicy -Scope /subscriptions/0c224eaf-e4fb-47d9-80ba-d44bacff6ffb/resourceGroups/ARMPolicies

```
PS C:\> New-AzureRmPolicyAssignment -Name locationPolicyAssignment -PolicyDefinition $locationpolicy -Scope /subscriptions/0c224eaf-e4fb-47d9-80ba-d44bacff6ffb/resourceGroups/ARMPolicies

Name                : locationPolicyAssignment
ResourceId          : /subscriptions/0c224eaf-e4fb-47d9-80ba-d44bacff6ffb/resourceGroups/ARMPolicies/providers/Microsoft.Authorizatio
                      n/policyAssignments/locationPolicyAssignment
ResourceName        : locationPolicyAssignment
ResourceType        : Microsoft.Authorization/policyAssignments
ResourceGroupName   : ARMPolicies
SubscriptionId      : 0c224eaf-e4fb-47d9-80ba-d44bacff6ffb
Properties          : @{policyDefinitionId=/subscriptions/0c224eaf-e4fb-47d9-80ba-d44bacff6ffb/providers/Microsoft.Authorization/poli
                      cyDefinitions/regionPolicyDefinition; scope=/subscriptions/0c224eaf-e4fb-47d9-80ba-d44bacff6ffb/resourceGroups/
                      ARMPolicies}
PolicyAssignmentId  : /subscriptions/0c224eaf-e4fb-47d9-80ba-d44bacff6ffb/resourceGroups/ARMPolicies/providers/Microsoft.Authorizatio
                      n/policyAssignments/locationPolicyAssignment
```
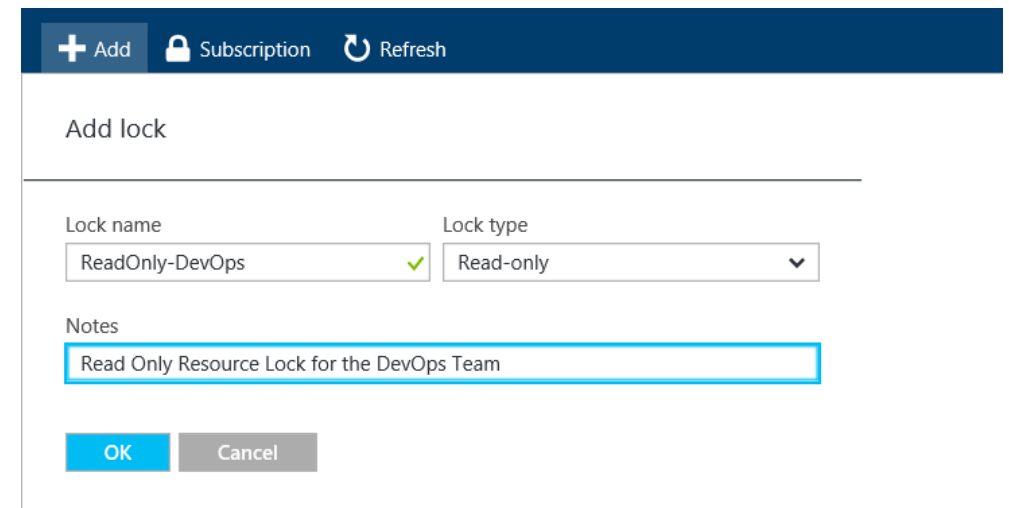
Microsoft

Resource Locks

Microsoft Services

Resource Locks

- ARM Resource Locks allow you to prevent the accidental deletion or modification of resource groups or resources in your subscription.

- There are 2 resource lock levels: Delete or ReadOnly.
  - Delete means authorized users can still read and modify a resource, but they can't delete it.
  - ReadOnly means authorized users can read from a resource, but they can't delete it or perform any actions on it.

- Applies to Everyone including Administrators.

Resource Lock Management

- Create resource locks using the portal, ARM template, PowerShell or REST API.

- Applied at the subscription, resource group or resource level.

- When a lock is applied at a parent scope, all child resources inherit the same lock, even resources you add later inherit the lock from the parent.

- Owner and User Access Administrator can create and delete resource locks.

Lab: Management Policies & Resource Locks

Microsoft Services

Sensitive Information & Templates

- Sensitive information such as VM secrets, certificates and network routing information should not be specified in an ARM template.

- Use Azure Key Vault with Resource Manager to orchestrate and securely store VM secrets and certificates.

- Using Key Vault means that the ARM template references a URI that contains the secrets.

- The loading of secrets into a VM at deployment occurs via direct channel between the Azure Fabric and the Key Vault within the confines of the Microsoft datacenter.

- Maintain separate templates for vault creation and VM deployment.

Service Principals for Cross Subscription Access

- Use service principals with role based access control to restrict permission for cross subscription access e.g. a cloud service provider accessing a customer subscription.

- Scope access to a specific resource group or resource e.g. a storage account.

- Grant the most restrictive permission required e.g. read only access.

- Enable auditing on resources that are accessed.

- Use organizational accounts for more control.

Network Information & Templates

- Many scenarios will have requirements that specify how traffic to one or more VM instances in your virtual network is controlled.

- Use a Network Security Group (NSG) to define this part of the ARM template.

- NSG's control all inbound and outbound traffic to a NIC or subnet as opposed to an endpoint based ACL which only works on the public port that is exposed.

- A NIC or subnet can be associated with only 1 NSG and each NSG can contain up to 200 rules.

wap2510	Network interface	West Europe		...
CBADFS	Network security group	West Europe		...
ADFS1PIP	Public IP address	West Europe		...

Demo: Management
Policies & Resource Locks
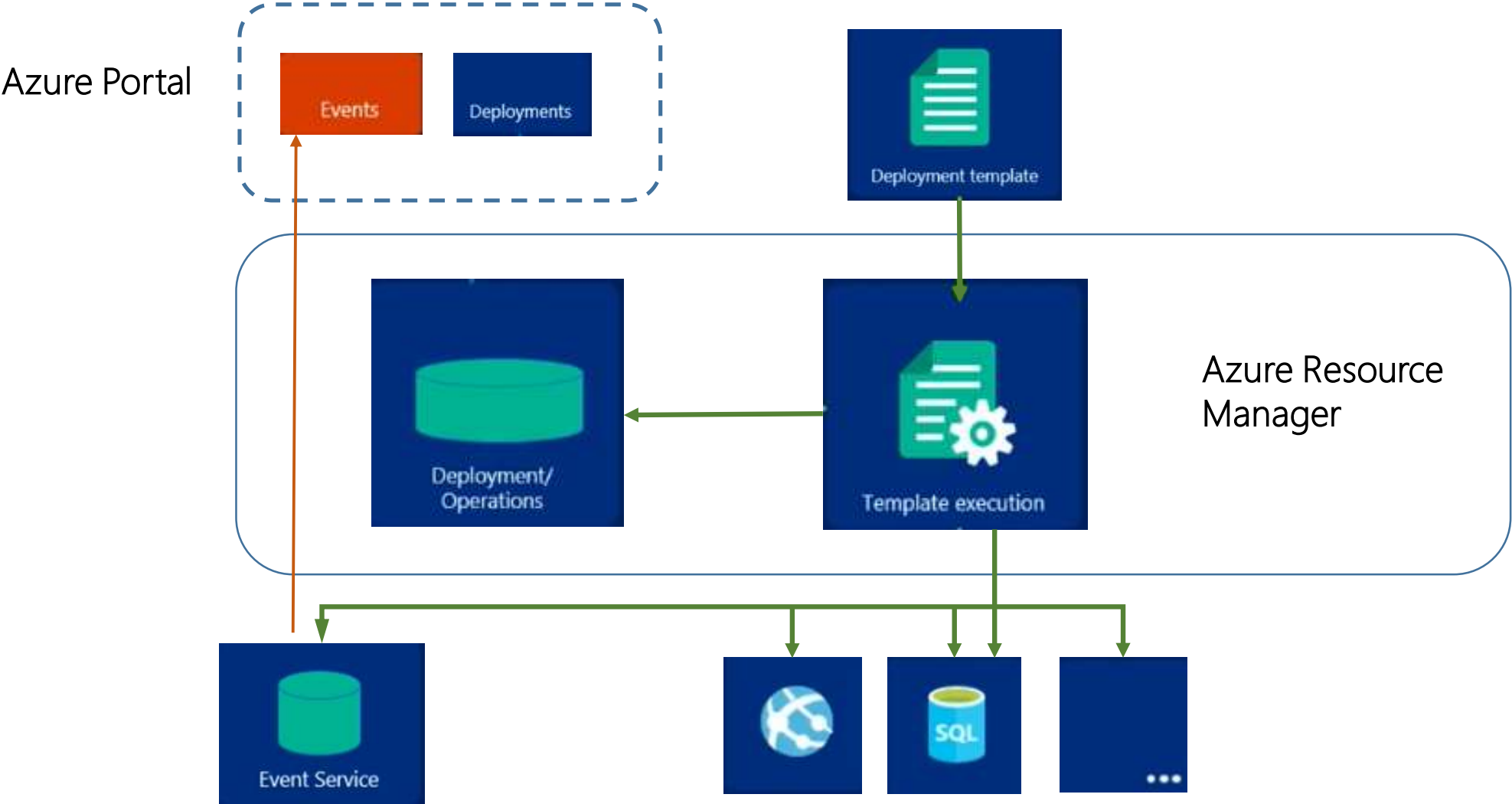
Auditing & Troubleshooting

Microsoft Services

Event Logging Process

- As resources are being deployed, the resource providers will interact with the Azure Event service.

- Event service gathers event information about what is happening behind the scenes.

- As the template execution is taking place, information is being requested from the resource providers, which in turn, are reporting all actions they are taking to the Event service.

- The Event service then notifies the event viewer of operations which are logged and made viewable in the Audit Logs blade.

- Includes all activities that are performed.

Resource Deployment Flow

Microsoft

Activity Logs
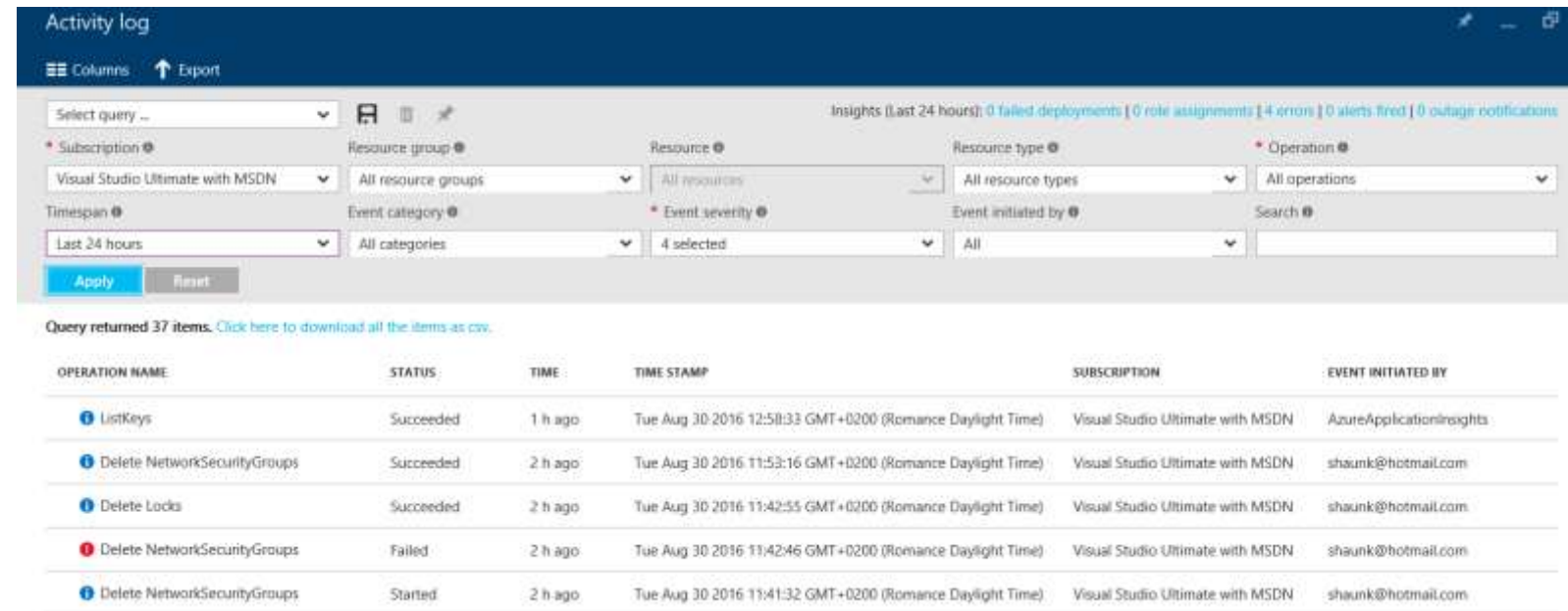
Microsoft Services

Activity Logs

- Use activity logs to find an error when troubleshooting or to monitor how a user in your organization modified a resource.

- Using activity logs, you can determine:
  - What operations were taken on the resources in your subscription.
  - Who initiated the operation (although operations initiated by a backend service do not return a user as the caller).
  - When the operation occurred.
  - The status of the operation.
  - The values of other properties that might help you research the operation.

- The activity log contains all write operations (PUT, POST & DELETE) performed on your resources and does not include read operations (GET).

View Activity Logs

- View activity logs using Azure portal, PowerShell, Azure CLI or REST API.

- Activity logs are retained for 90 days but can only be queried for 15 days or less.

- Use Get-AzureRmLog -ResourceGroup <ResourceGroupName> to view activity logs using PowerShell.

Troubleshooting

Microsoft Services

Deployment Operation Logs

- Deployment operations are logged under 2 areas, deployment operation logs and activity logs.

- Deployment operation logs include a detailed recording of the deployment and provide enough information to begin troubleshooting.

- Activity logs include basic summary logging and also more detailed logging using JSON.

- Correlation ID's are used to track related events.

View Deployment Operation Logs

- View deployment logs using Azure portal, PowerShell, Azure CLI or REST API.
- Use **Events** to view all related operations for a deployment and locate the failed component.
- Use Get-AzureRmResourceGroupDeployment –ResourceGroup <ResourceGroupName> to get the overall status of a deployment.
- Use Get-AzureRmResourceGroupDeploymentOperation –ResourceGroupName <ResourceGroupName> to view deployment logs for each operation in a deployment.
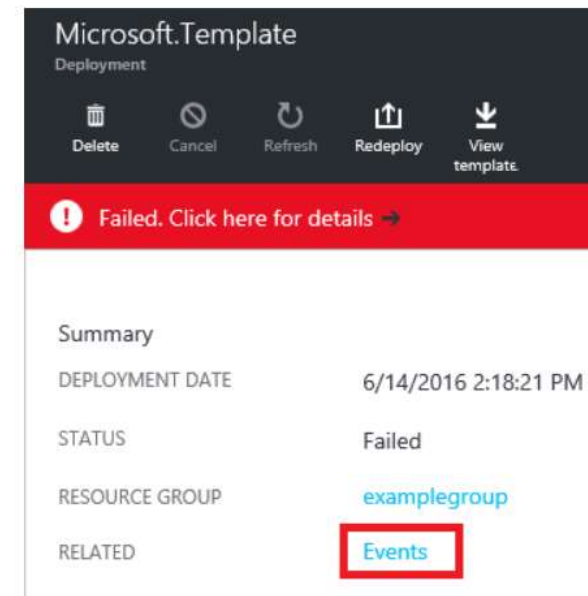
Common Deployment Errors

- Some of the most common deployment errors today include:

  - Invalid template or resource

  - Incorrect segment lengths

  - Resource name taken or is in use by another resource

  - Cannot find resource during deployment

  - Could not find member 'copy' on object

  - Regex/casing problems in names

  - SKU not available

  - No registered provider found

  - Quota exceeded

  - Authorization failed

  - Case sensitive URIs

  - Storage account provisioning delay

  - DSC Errors

Invalid Template or Resource

Cause:

This error commonly indicates that there is a syntax error in the template.

```
Code=InvalidTemplate

Message=Deployment template validation failed
```

Resolution:

When you receive this type of error, carefully review the expression syntax. Consider using a JSON editor like Visual Studio or Visual Studio Code which can warn you about syntax errors.

Incorrect Segment Lengths

Cause:

This error commonly indicates that the resource name is not in the correct format.

```
Code=InvalidTemplate                                              Copy 📋
Message=Deployment template validation failed: 'The template resource {resource-name}
for type {resource-type} has incorrect segment lengths.
```

Resolution:

When you receive this type of error, carefully review the resource name and ensure that it has one less segment than the resource type e.g. the resource name "myHostingPlanName" of type "Microsoft.Web/serverfarms" is correct as opposed to the resource name "appPlan/myHostingPlanName" of type "Microsoft.Web/serverfarms".

Resource name taken or is in use by another resource

Cause:

This error commonly indicates that the name is not unique across all of Azure.

```
Code=StorageAccountAlreadyTaken                          Copy
Message=The storage account named mystorage is already taken.
```

Resolution:

When you receive this type of error, review the resource name and provide a name that is unique across all of Azure.

Cannot find resource during deployment

Cause:

This error commonly indicates that the supporting resource does not exist.

Resolution:

When you receive this type of error, ensure that the supporting resource exists prior to the dependent resource e.g. an App Service plan is a supporting resource for a web app and must exist prior to the web app being created. Use the **dependsOn** element in your template to define this.

Could not find member 'copy' on object

Cause:

This error commonly indicates that you have applied the copy element to a part of the template that does not support this element.

Resolution:

When you receive this type of error, carefully review the template and remove the copy element from any resource type properties since the copy element can only be applied to resource types and not their properties.

Regex/Casing problems in names

Cause:

This error indicates that the specified name does not conform to an allowed regular expression or regex i.e. lowercase, numbers and hyphens.

Resolution:

When you receive this type of error, review the name and ensure that it conforms to an allowed regular expression e.g. using contoso.com instead of contoso*com when creating a DNS zone.

SKU not available

Cause:

This error commonly indicates that the resource SKU that you have selected (such as VM size) is not available for the location you have selected.

```
Code: SkuNotAvailable
Message: The requested tier for resource '<resource>' is currently not available in
```
`Copy 📋`

Resolution:

When you receive this type of error, log into the Azure portal and begin adding a new resource through the UI. As you set the values, you will see the available SKUs for that resource. Or contact Azure Customer Support.

No registered provider found

Cause:

This error commonly indicates one of three reasons, the location is not supported for the resource type, the API version is not supported for the resource type or the resource provider has not been registered for your subscription.

```
Code: NoRegisteredProviderFound          Copy 📋
Message: No registered resource provider found for location '<location>' and API ver:
```

Resolution:

The error message should give you suggestions for the supported locations and API versions. You can change your template to one of the suggested values.

Quota exceeded

Cause:

This error commonly indicates that the deployment exceeds a quota, which could be per resource group, subscriptions, accounts, and other scopes.

```
statusCode:Conflict
serviceRequestId:xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx
statusMessage:{"error":{"code":"OperationNotAllowed","message":"Operation results in exceeding quota limits of Core. Maximum allowed: 4, Current in use: 4, Additional requested: 2."}}
```

Resolution:

Check Azure subscription and service limits, quotas, and constraints to confirm the maximum quota for your resource, raise a support case to have the quota increased if it is not at it's maximum limit.

Authorization failed

Cause:

You may receive this error during deployment because the account or service principal attempting to deploy the resources does not have access to perform those actions.

Resolution:

When you receive this type of error, check the permissions of the account being used and also confirm that there are no limits that are enforced by any Management Policies.

Case sensitive URIs

Cause:

This issue indicates that the specified URI parameter value does not conform to an allowed casing type.

Resolution:

When you experience this type of issue, review the URI parameter value and ensure that it conforms to an allowed casing type e.g. dsc.zip is not the same as DSC.zip.

Storage account provisioning delay

Cause:

This issue indicates that the recreation of a recently deleted storage account has failed.

Resolution:

When you experience this type of issue, wait approximately 10 minutes and recreate your storage account.

DSC Errors

Cause:

There could be a number of different reasons for issues that are experienced during deployment using PowerShell DSC.

Resolution:

When you experience PowerShell DSC issues, review the log files located at C:\Packages\Plugins\Microsoft.Powershell.DSC\1.7.0.0\ on the problem virtual machine.

Know when a deployment is ready

- Azure Resource Manager reports success on a deployment but this does not mean that the deployment is ready for use.

- Other tasks may still need to complete before the deployment is ready e.g. waiting for updates or running post creation scripts.

- Azure can be prevented from reporting deployment success by using a custom script to monitor the deployment and report success once all tasks have been successfully completed.

- Use the dependsOn element in a template to specify dependencies.

Demo: Activity & Deployment Logs