


DevOps

Week 05


Murtaza Munawar Fazal


Azure Portal


- The Azure portal lets you build, manage, and monitor everything from simple web apps to complex cloud applications in a single, unified console.
 - Search resources, services, and docs.
 - Manage resources.
 - Create customized dashboards and favorites.
 - Access the Cloud Shell.
 - Receive notifications.
 - Links to the Azure documentation.


 Search resources, services, and docs (G+/)


Azure services


Create a resource




Storage accounts


Subscriptions


Activity log


Network Watcher

Recent resources

Name	Type	Last Viewed
 vault135	Recovery Services vault	22 hours ago
 RSV-Backup	Recovery Services vault	22 hours ago

Azure Cloud Shell

- Azure Cloud Shell is an interactive, browser-accessible shell for managing Azure resources. It provides the flexibility of choosing the shell experience that best suits the way you work. Linux users can opt for a Bash experience, while Windows users can opt for PowerShell.
- Cloud Shell enables access to a browser-based command-line experience built with Azure management tasks in mind. You can use Cloud Shell to work untethered from a local machine in a way only the cloud can provide.



Azure Cloud Shell - Features

- Is temporary and requires a new or existing Azure Files share to be mounted.
- Offers an integrated graphical text editor based on the open-source Monaco Editor.
- Authenticates automatically for instant access to your resources.
- Runs on a temporary host provided on a per-session, per-user basis.
- Requires a resource group, storage account, and Azure File share.
- Uses the same Azure file share for both Bash and PowerShell.
- Permissions are set as a regular Linux user in Bash.

Azure PowerShell

- Azure PowerShell is a module that you add to Windows PowerShell or PowerShell Core to enable you to connect to your Azure subscription and manage resources. Azure PowerShell requires PowerShell to function. PowerShell provides services such as the shell window and command parsing. Azure PowerShell adds the Azure-specific commands.
- For example, Azure PowerShell provides the New-AzVm command that creates a virtual machine inside your Azure subscription. To use it, you would launch the PowerShell application and then issue a command.

Azure PowerShell

- Azure PowerShell is also available two ways:
 - inside a browser via the Azure Cloud Shell, or
 - with a local installation on Linux, macOS, or the Windows operating system.
- In both cases, you have two modes from which to choose:
 - you can use it in interactive mode in which you manually issue one command at a time, or
 - in scripting mode where you execute a script that consists of multiple commands.

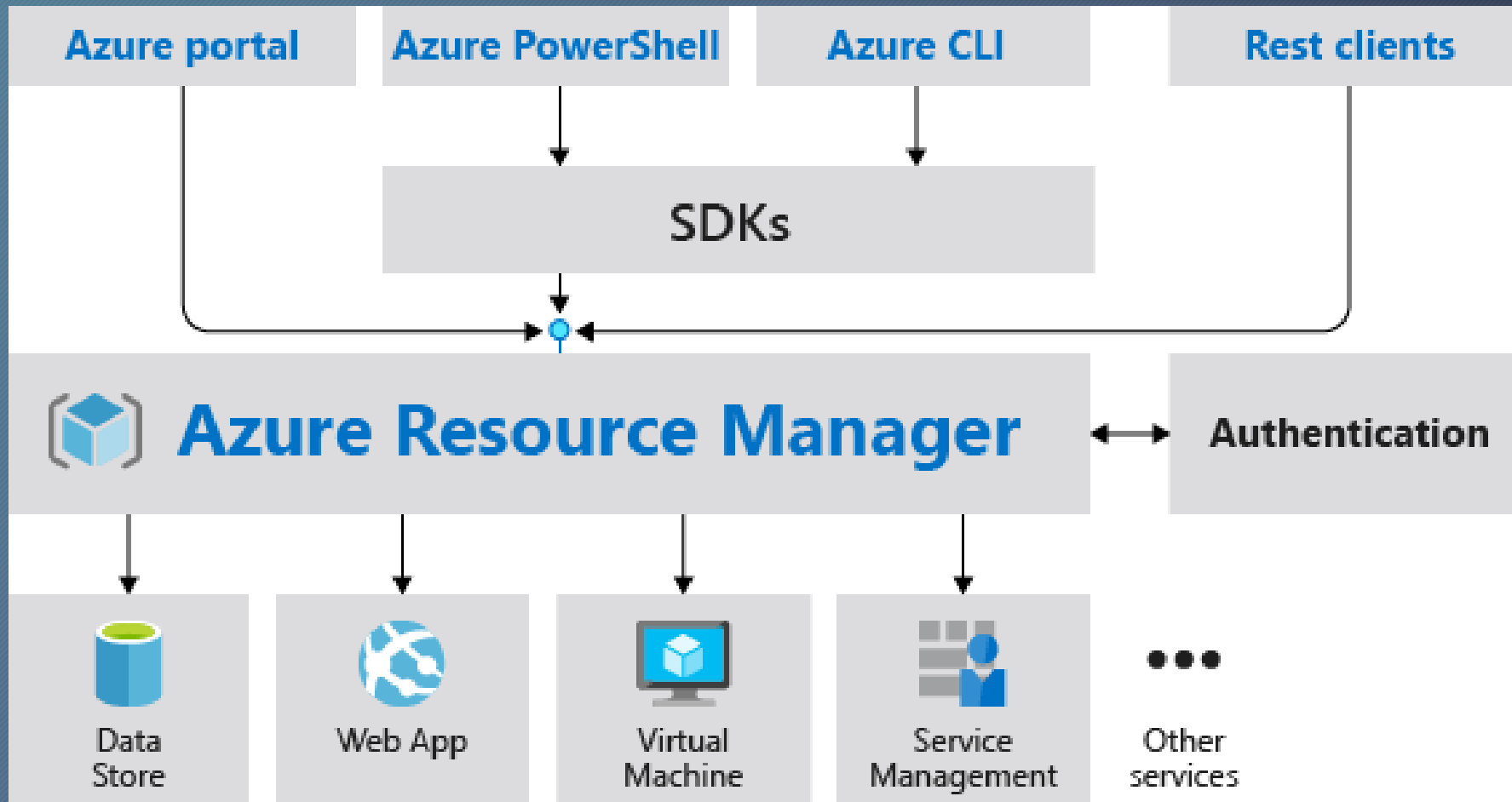
Azure CLI

- Azure CLI is a command-line program to connect to Azure and execute administrative commands on Azure resources. It runs on Linux, macOS, and Windows, and allows administrators and developers to execute their commands through a terminal, command-line prompt, or script instead of a web browser.
- Azure CLI provides cross-platform command-line tools for managing Azure resources. You can install the CLI locally on computers running the Linux, macOS, or Windows operating systems. You can also use Azure CLI from a browser through Azure Cloud Shell.

Azure Resource Manager

- 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. You use a template for deployment and that template can work for different environments such as testing, staging, and production. Azure Resource Manager provides security, auditing, and tagging features to help you manage your resources after deployment.
- Azure Resource Manager provides a consistent management layer to perform tasks through Azure PowerShell, Azure CLI, Azure portal, REST API, and client SDKs.

Azure Resource Manager



Azure Resource Manager

- **Resource** - A manageable item that is available through Azure. Some common resources are a virtual machine, storage account, web app, database, and virtual network, but there are many more.
- **Resource Group** - A container that holds related resources for an Azure solution. The resource group can include all the resources for the solution, or only those resources that you want to manage as a group. You decide how you want to allocate resources to resource groups based on what makes the most sense for your organization.
- **Resource Provider** - A service that supplies the resources you can deploy and manage through Resource Manager. Each resource provider offers operations for working with the resources that are deployed. Some common resource providers are Microsoft.Compute, which supplies the virtual machine resource, Microsoft.Storage, which supplies the storage account resource, and Microsoft.Web, which supplies resources related to web apps.
- **Template** - A JavaScript Object Notation (JSON) file that defines one or more resources to deploy to a resource group. It also defines the dependencies between the deployed resources. The template can be used to deploy the resources consistently and repeatedly.
- **Declarative Syntax** - Syntax that lets you state "Here is what I intend to create" without having to write the sequence of programming commands to create it. The Resource Manager template is an example of declarative syntax. In the file, you define the properties for the infrastructure to deploy to Azure.

Resource group

- A resource group is a container that holds related resources for an Azure solution. The resource group can include all the resources for the solution, or only those resources that you want to manage as a group. You decide how you want to allocate resources to resource groups based on what makes the most sense for your organization. Generally, add resources that share the same lifecycle to the same resource group so you can easily deploy, update, and delete them as a group.
- The resource group stores metadata about the resources. Therefore, when you specify a location for the resource group, you are specifying where that metadata is stored. For compliance reasons, you may need to ensure that your data is stored in a particular region.

Template

- An Azure Resource Manager template precisely defines all the Resource Manager resources in a deployment. You can deploy a Resource Manager template into a resource group as a single operation.
- Using Resource Manager templates will make your deployments faster and more repeatable. For example, you no longer have to create a VM in the portal, wait for it to finish, and then create the next VM. Resource Manager template takes care of the entire deployment for you.

Template Benefits

- **Templates improve consistency.** Resource Manager templates provide a common language for deployments. Regardless of the tool or SDK that you use to deploy the template, the structure, format, and expressions inside the template remain the same.
- **Templates help express complex deployments.** Templates enable you to deploy multiple resources in the correct order. For example, you wouldn't want to deploy a virtual machine prior to creating an operating system (OS) disk or network interface. Resource Manager maps out each resource and its dependent resources and creates dependent resources first. Dependency mapping helps ensure that the deployment is carried out in the correct order.
- **Templates reduce manual, error-prone tasks.** Manually creating and connecting resources can be time consuming, and it's easy to make mistakes. Resource Manager ensures that the deployment happens the same way every time.
- **Templates are code.** Templates express your requirements through code. Think of a template as a type of Infrastructure as Code that can be shared, tested, and versioned similar to any other piece of software. Also, because templates are code, you can create a "paper trail" that you can follow. The template code documents the deployment.

Template Benefits

- **Templates promote reuse.** Your template can contain parameters that are filled in when the template runs. A parameter can define a username or password, a domain name, and so on. Template parameters enable you to create multiple versions of your infrastructure, such as staging and production, while still using the exact same template.
- **Templates are linkable.** You can link Resource Manager templates together to make the templates themselves modular. You can write small templates that each define a piece of a solution, and then combine them to create a complete system.
- **Templates simplify orchestration.** You only need to deploy the template to deploy all of your resources. Normally this would take multiple operations.

Azure Resource Manager template schema

- Azure Resource Manager templates are written in JSON, which allows you to express data stored as an object (such as a virtual machine) in text. A JSON document is essentially a collection of key-value pairs. Each key is a string, whose value can be:
 - A string
 - A number
 - A Boolean expression
 - A list of values
 - An object (which is a collection of other key-value pairs)
- A Resource Manager template can contain sections that are expressed using JSON notation, but aren't related to the JSON language itself:

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

Azure Resource Manager template schema

- **\$schema:**
 - Location of the JSON schema file that describes the version of the template language. Use the URL shown in the preceding example.
- **ContentVersion:**
 - Version of the template (such as 1.0.0.0). You can provide any value for this element. Use this value to document significant changes in your template. This value can be used to make sure that the right template is being used.
- **Parameters**
 - Values that are provided when deployment is executed to customize resource deployment.
- **Variables**
 - Values that are used as JSON fragments in the template to simplify template language expressions.
- **Functions**
 - User-defined functions that are available within the template.
- **Resources**
 - Resource types that are deployed or updated in a resource group.
- **Outputs**
 - Values that are returned after deployment.

Bicep Templates

- Azure Bicep is a domain-specific language (DSL) that uses declarative syntax to deploy Azure resources. It provides concise syntax, reliable type safety, and support for code reuse.
- You can use Bicep instead of JSON to develop your Azure Resource Manager templates (ARM templates). The JSON syntax to create an ARM template can be verbose and require complicated expressions. Bicep syntax reduces that complexity and improves the development experience. Bicep is a transparent abstraction over ARM template JSON and doesn't lose any of the JSON template capabilities.

Demo

Generating a reusable Template for Virtual Machine