Microsoft

Bicep Upskilling Workshop

Lab 1
Introduction to Bicep

DISCLSIMER

© 2025 Microsoft Corporation. All rights reserved. Microsoft, Windows and other product names are or may be registered trademarks and/or trademarks in the U.S. and/or other countries.

The information herein is for informational purposes only and represents the current view of Microsoft Corporation or any Microsoft Group affiliate as of the date of this presentation. Because Microsoft must respond to changing market conditions, it should not be interpreted to be a commitment on the part of Microsoft, and Microsoft cannot guarantee the accuracy of any information provided after the date of this presentation.

MICROSOFT MAKES NO WARRANTIES, EXPRESS, IMPLIED OR STATUTORY, AS TO THE INFORMATION IN THIS PRESENTATION.

CONFIDENTIALITY

© 2025 Microsoft Corporation. All rights reserved. Any use or distribution of these materials without express authorization of Microsoft Corp. is strictly prohibited.

Microsoft

Contents

Microsoft

Lab 1 - Introduction to Bicep

Objective(s)	• To be able to define Azure resources in Bicep • To be able to deploy Azure resources using Bicep template • To be able to use Bicep module • To understand parameters and variables • To be able to use parameters and variables
Duration of Lab	• 1h
Prerequisite(s)	• Azure Subscription and Resource Group • **Contributor** Role in a selected Resource Group • Visual Studio Code installed • Bicep VS Code Extension installed • Azure CLI installed
Tool(s)	• VS Code • Azure Portal • Azure CLI
Exercises	1. Create your first Bicep template for Storage Account 2. Deploy your Bicep template to Azure using Azure CLI
Subscription	[Selected Subscription]
Resource Group	[Selected RG]
Navigation	Throughout this Lab, we will open and use several Browser tabs for easy access. Until the end of the Lab, keep your Browser tabs open.

Naming Convention for Labs

For completing various labs during the workshop, we will use this naming convention. It is slightly different from Microsoft online guidance (Define your naming convention - Cloud Adoption Framework | Microsoft Learn).

The naming convention below is designed to group your Azure resources together for easy access.

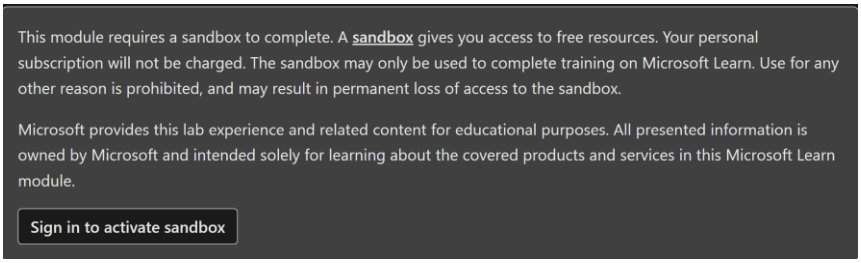
[you name/initials]-[azure service short name/acronym]-[service description]

Exercise 1- Create Your First Bicep Template (15 min)

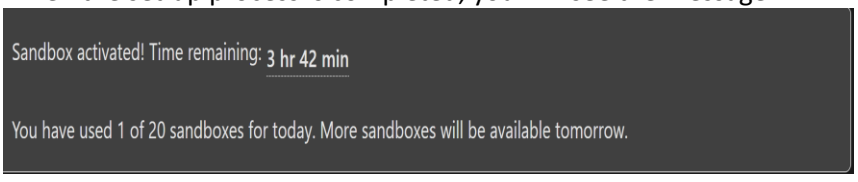
Topics	In this exercise, we will cover the following topics. • Define Azure resources in Bicep template • Create Bicep template in VS Code • Deploy Bicep template to Azure using Azure CLI • Use Parameters and Variables
Duration	• 15 min
Tool(s)	• Visual Studio Code • Azure portal • Azure CLI
Lab Scenario	We will use Microsoft sandbox for this exercise. Throughout the modules, use the online training module - Exercise - Define resources in a Bicep template - Training | Microsoft Learn. You can complete the exercises locally without using MS Sandbox. If you prefer working locally, skip the *Prerequisites* section
Subscription	[selected subscription]
Resource Group	[selected RG]

Prerequisites

1. Open the online training module page in your web browser
2. Activate sandbox for the lab



3. Complete the sign-up process (e.g. setting up authentication)
4. When the set-up process is completed, you will see the message

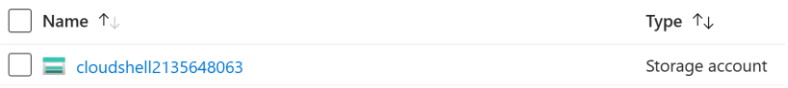


Note: The first time you activate a sandbox and accept the terms, your Microsoft account is associated with a new Azure directory named ***Microsoft Learn Sandbox***. You're also added to a special subscription named ***Concierge Subscription***.
The sandbox environment is temporary created for your use, only for **4h** and 20 sandboxes per day

5. Open new browser tab and go to *Azure portal* (portal.azure.com)
6. You need to change the directory for accessing the sandbox environment
7. Click the **Settings** icon  at the top right of the portal. Available directories are displayed
8. Select *Microsoft Learn Sandbox* by clicking the **Switch** button

9. Your will be redirected to the sandbox directory
10. A new *Resource Group* is created for you. It looks like this:

11. Select the **Resource Group**. At the moment, one Storage Account is provisioned. The Storage Account is primarily created for *Cloud Shell* in Azure portal



Module 1 – Create Bicep template for Storage Account

1. Go to Azure portal (portal.azure.com) and select your *Subscription* and the *Resource Group*
2. Open a new browser tab and go to the online training module (Exercise - Define resources in a Bicep template - Training | Microsoft Learn)
3. Follow the instructions on *"Create a Bicep template that contains a storage account"*
4. Follow the instructions on *"Deploy the Bicep template to Azure"*
5. Follow the instructions on *"Verify the deployment"*

Module 2 – Update Bicep template with additional resources

1. Continue on the instructions on *"Add an App Service plan and app to your Bicep template"*
2. Once deployment is completed, verify that your deployment is successful
   Note: If you want to change the *Azure region* to Australia East, replace the value with **"australiaeast"**

Module 3 – Add Parameters and Variables

1. Move onto the next module by selecting the **Next** button for *"Add flexibility by using parameters and variables"*
2. Skim through the *"Parameters"*, *"Variables"* and *"Expressions"* sections in the page (just spend about 5min and do the rest at your own time)
3. Scroll down to the bottom and select the **Next** button for *"Exercise - Add parameters and variables to your Bicep template"* (Exercise - Add parameters and variables to your Bicep template - Training | Microsoft Learn)
4. Follow the instructions on this page

Summary

ACHIEVEMENTS

After you have completed the Lab, you are now able to:

- ✓ Create Bicep template in VS Code
- ✓ Create Azure resources using Bicep
- ✓ Use *intellisense* in VS Code to change or update script
- ✓ Deploy Bicep template to Azure using Azure CLI
- ✓ Use Parameters and Variables for Bicep

Exercise 2- Use Modules (15 min)

Topics	In this exercise, we will cover the following topics. • Group related resources by using modules • Work with Parameters • Secure Parameters
Duration	• 15 min
Tool(s)	• Azure portal
Lab Scenario	Because you've declared your resources in a template file, you can quickly deploy the resources for new toy launches without needing to manually configure resources in the Azure portal. The IT manager can see your Bicep code is becoming more complex and has an increasing number of resources defined, so they've asked if you can make the code more ***modularised***.
Subscription	[selected subscription]
Resource Group	[selected RG]

Module 1 – Create modules

1. Read through or skim through the articles on Group related resources by using modules - Training | Microsoft Learn
2. Proceed to the Next by clicking the **Next** button
3. Follow the instructions on Exercise - Refactor your template to use modules - Training | Microsoft Learn
4. Verify your deployment

Module 2 – Work with Parameters

1. Read through or skim through the article on Parameters - Understand parameters - Training | Microsoft Learn
2. Add parameter description in you *main.bicep* file like below

```
@description('The Azure region into which the resources should be deployed.')
param location string = 'eastus'
```

3. Read through or skim though the article on Parameter file - Provide values using parameter files - Training | Microsoft Learn

Module 3 – Secure Parameters

1. Read through or skim through the article on securing parameters - Secure your parameters - Training | Microsoft Learn
2. Follow the instructions on Exercise - Add a parameter file and secure parameters - Training | Microsoft Learn

Summary

ACHIEVEMENTS

After you have completed the Lab, you are now able to:

- ✓ Create Bicep module(s)
- ✓ Work with parameters
- ✓ Understand how to secure parameters by using the attribute '@secure()'
- ✓ Use Key Vault to store secrets as well as sensitive information, and retrieve the value within your Bicep file

Exercise 3 – Create Parameter Files (30min)

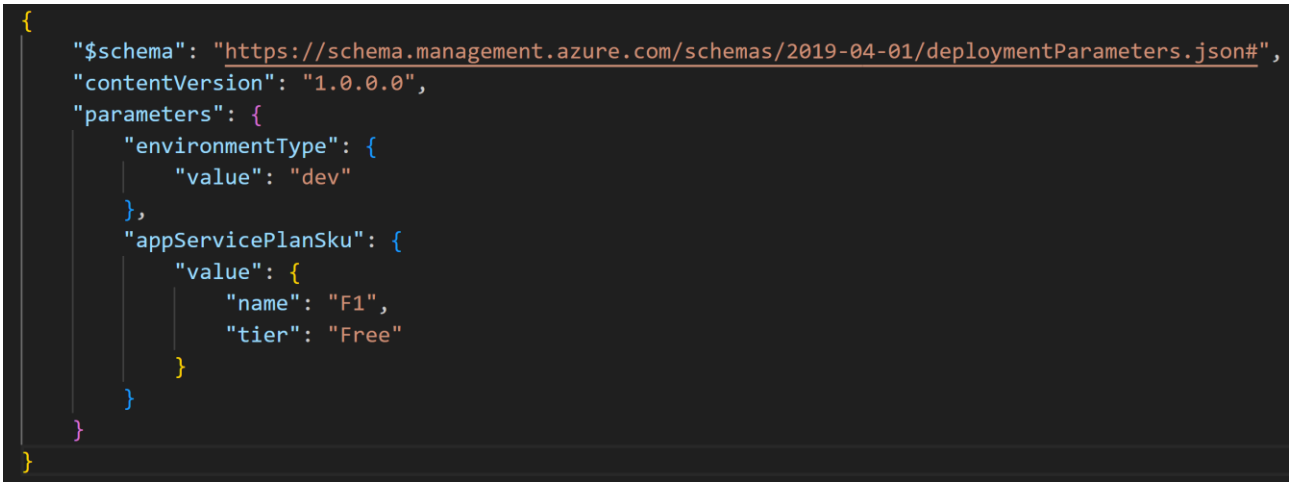
Prerequisite(s)	• Azure Table (prepared by MS instructor)
Topics	In this exercise, we will cover the following topics. • Create parameter file (json) • Update Bicep template to support the parameter file • Deploy Bicep and Parameter templates to Azure using Azure CLI
Duration	• 30 min
Subscription	[selected subscription]
Resource Group	[selected RG]

Module 1 – Create parameter file (.json)

1. In VS Code, add a new file called ***main.parameters.dev.json*** file
2. In the *main.parameters.dev.json*, add the following code:

```
{
  "$schema": "https://schema.management.azure.com/schemas/2019-04-01/deploymentParameters.json#",
  "contentVersion": "1.0.0.0",
  "parameters": {
    "environmentType": {
      "value": "dev"
    },
    "appServicePlanSku": {
      "value": {
        "name": "F1",
        "tier": "Free"
      }
    }
  }
}
```

It looks like this:



Module 2 – Update and tidy up main Bicep template for parameter file

1. We need to fix *bicep.main* and *appService.main* files
2. In ***main.bicep***, add a new parameter called **appServicePlanSku** in object

```
param appServicePlanSku object
```

3. Change the environment types to include **'dev', 'test'**, and **'prod'** instead

```
@allowed([
  'dev'
  'test'
  'prod'
])
param environmentType string
```

4. In *main.bicep*, remove the existing parameter, *environmentType*, and add the newly created parameter, *appServicePlanSku*, in AppService module statement

```
module appService './appService-ex3.bicep' = {
  name: 'appService'
  params: {
    location: location
    appServiceAppName: appServiceAppName
    appServicePlanSku: appServicePlanSku
  }
}
```

5. In ***appService.bicep***, remove unused parameter, *environmentType*
6. Instead, add a parameter, appServicePlanSku (object)

```
@description('The name and tier of the App Service plan SKU.')
param appServicePlanSku object
```

7. Update the sku definition for AppService like this

```
resource appServicePlan 'Microsoft.Web/serverfarms@2024-04-01' = {
  name: appServicePlanName
  location: location
  sku: appServicePlanSku
}
```

8. Previously, sku definition looks like this

```
    sku: {
      name: appServicePlanSkuName
    }
```

9. Previously, we were specifying the *'name' property* of the *SKU object*. But now, we pass the *'object'* value passed down from *main.bicep*
10. *main.bicep* template looks like this

```
@description('The Azure region into which the resources should be deployed.')
param location string = 'australiaeast'

param storageAccountName string = 'statoy${uniqueString(resourceGroup().id)}'
param appServiceAppName string = 'kuext-web-toy${uniqueString(resourceGroup().id)}'

@allowed([
  'dev'
  'test'
  'prod'
])
param environmentType string

param appServicePlanSku object

var storageAccountSkuName = (environmentType == 'prod') ? 'Standard_GRS' : 'Standard_LRS'

resource storageAccount 'Microsoft.Storage/storageAccounts@2023-05-01' = {
  name: storageAccountName
  location: location
  sku: {
    name: storageAccountSkuName
  }
  kind: 'StorageV2'
  properties: {
    accessTier: 'Hot'
  }
}
module appService './appService-ex3.bicep' = {
  name: 'appService'
  params: {
    location: location
    appServiceAppName: appServiceAppName
    appServicePlanSku: appServicePlanSku
  }
}

output appServiceAppHostName string = appService.outputs.appServiceAppHostName
```

11. *appService.bicep* template looks like this

```
param location string
param appServiceAppName string

@description('The name and tier of the App Service plan SKU.')
param appServicePlanSku object

var appServicePlanName = 'kuext-toy-launch-plan'

resource appServicePlan 'Microsoft.Web/serverfarms@2024-04-01' = {
  name: appServicePlanName
  location: location
  sku: appServicePlanSku
}

resource appServiceApp 'Microsoft.Web/sites@2024-04-01' = {
```

…

12. *main.parameters.dev.json* looks like this

```
{
    "$schema": "https://schema.management.azure.com/schemas/2019-04-01/deploymentParameters.json#",
    "contentVersion": "1.0.0.0",
    "parameters": {
        "environmentType": {
            "value": "dev"
        },
        "appServicePlanSku": {
            "value": {
                "name": "F1",
                "tier": "Free"
            }
        }
    }
}
```

Module 3 – Deploy with parameter file

1. Verify your bicep and parameter files by deploying it to Azure
2. In VS Code using the Terminal (already opened, and signed in to Azure), enter the command and run

```
az deployment group create \
  --name main \
  --template-file main.bicep \
  --parameters main.parameters.dev.json
```

3. Go back to Azure and your Resource Group
4. Check your deployment is successful

Summary

ACHIEVEMENTS

After you have completed the Lab, you are now able to:

- ✓ Create a parameter file (json) separately from main.bicep
- ✓ Define parameters based on environment-specific
- ✓ Deploy Bicep and parameter templates to Azure using Azure CLI