한국 마이크로소프트

Microsoft Technical Trainer

Enterprise Skills Initiative

Microsoft Azure

Azure Bicep

Microsoft

이 문서는 Microsoft Technical Trainer팀에서 ESI 교육 참석자분들에게 제공해 드리는 문서입니다.

요약

이 내용들은 표시된 날짜에 Microsoft에서 검토된 내용을 바탕으로 하고 있습니다. 따라서, 표기된 날짜 이후에 시장의 요구사항에 따라 달라질 수 있습니다. 이 문서는 고객에 대한 표기된 날짜 이후에 변화가 없다는 것을 보증하지 않습니다.

이 문서는 정보 제공을 목적으로 하며 어떠한 보증을 하지는 않습니다.

문서 작성 연혁

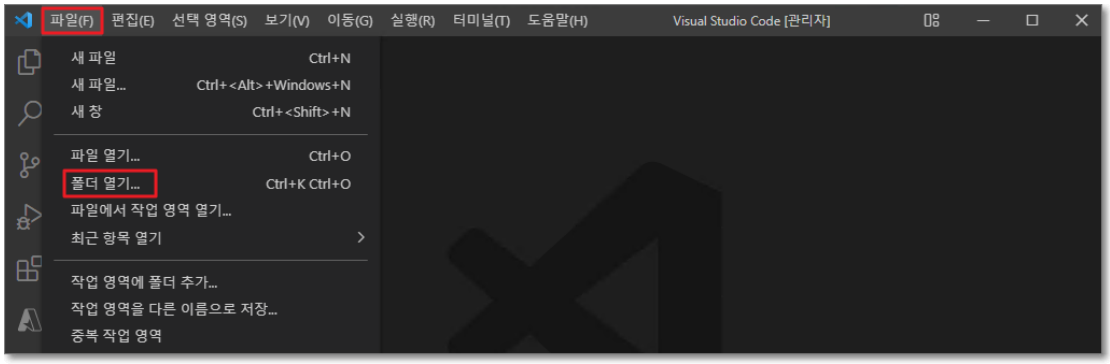
1. 이 문서는 2022년 03월 27일 한국 마이크로소프트(유), Microsoft Technical Trainer 팀 우진환에 작성되었습니다. (Version 1.0.0)

목차

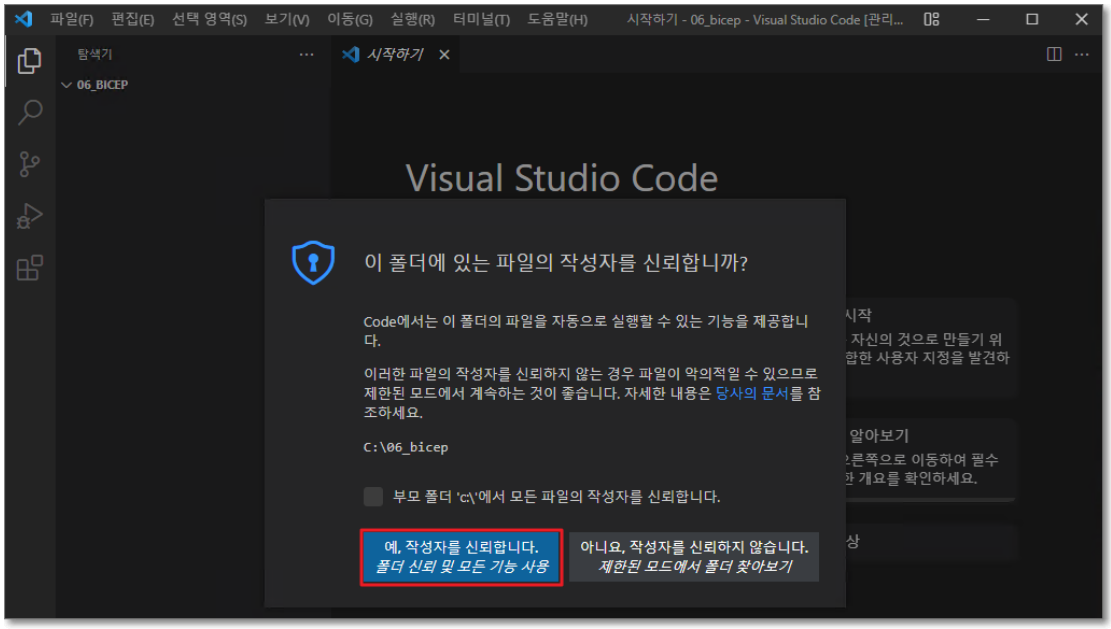
TASK 01. 매개 변수와 데코레이터 추가

이 작업에서는 Bicep을 사용하여 Azure App Service Plan 및 App Service 앱을 만듭니다. 또한 각 매개 변수에 데코레이터를 적용하여 예상한 값이 항상 포함되도록 구성합니다.

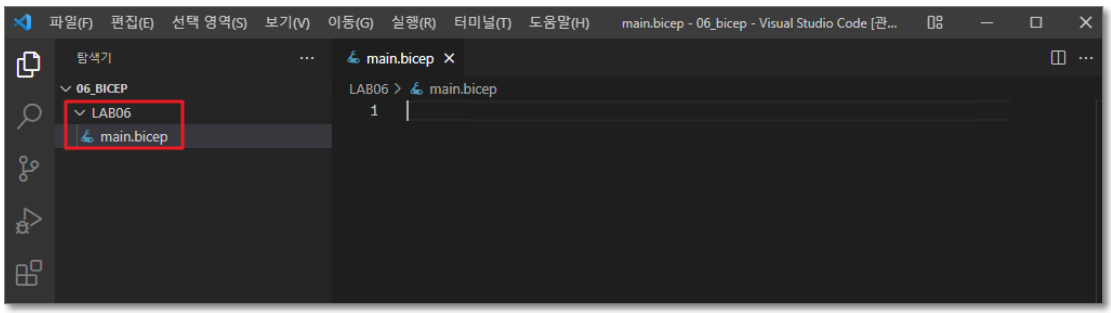
1. 실습 머신에 로그온한 후 [Visual Studio Code]를 실행합니다. 메뉴에서 [파일 - 폴더 열기]를 클릭합니다.



2. `C:\06_bicep` 폴더를 만들고 이 폴더를 선택합니다. [이 폴더에 있는 파일의 작성자를 신뢰합니까?] 창이 표시되면 "예, 작성자를 신뢰합니다"를 클릭합니다.



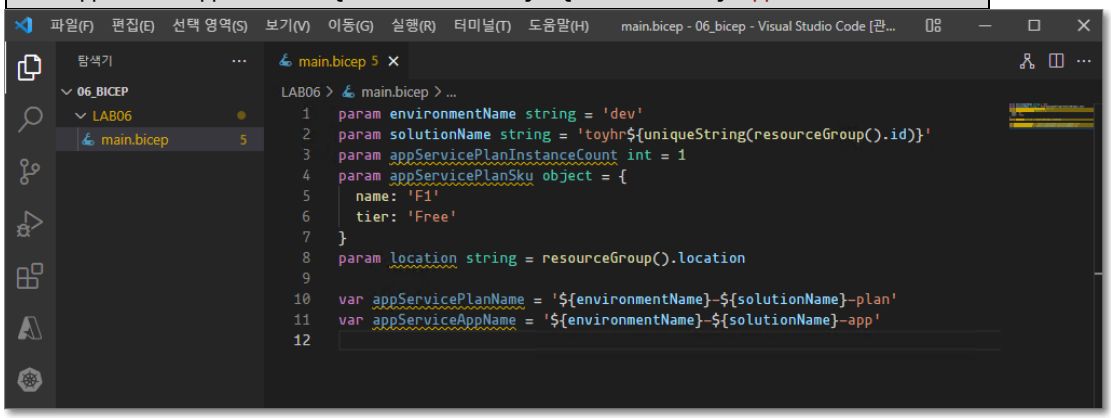
3. Visual Studio Code의 [탐색기]에서 "LAB06" 폴더를 만든 후 이 폴더에 "main.bicep" 파일을 만듭니다.



4. "main.bicep" 파일에 다음과 같은 코드를 작성합니다.
   - 여러 유형을 혼합하여 여러 매개 변수를 선언합니다. 각 매개 변수에는 기본값을 정의합니다.
   - 일부 값은 `string interpolation`, `uniqueString()` 함수, `resourceGroup()` 함수를 포함합니다.

- `uniqueString()` 함수는 글로벌하게 고유한 리소스 이름을 만드는데 유용하게 사용할 수 있습니다. `uniqueString()` 함수를 사용하면 동일한 리소스 그룹에 대한 배포에서는 이름이 동일하지만 다른 리소스 그룹이나 구독에 배포할 때는 다른 문자열을 반환합니다.
- Azure App Service Plan과 App Service 앱의 이름을 구성하는 변수를 정의합니다. 이 값에는 지정한 매개 변수의 일부가 포함됩니다.
- 매개 변수 값은 배포를 실행하는 사용자가 재정의할 수 있지만 변수는 재정의할 수 없습니다.

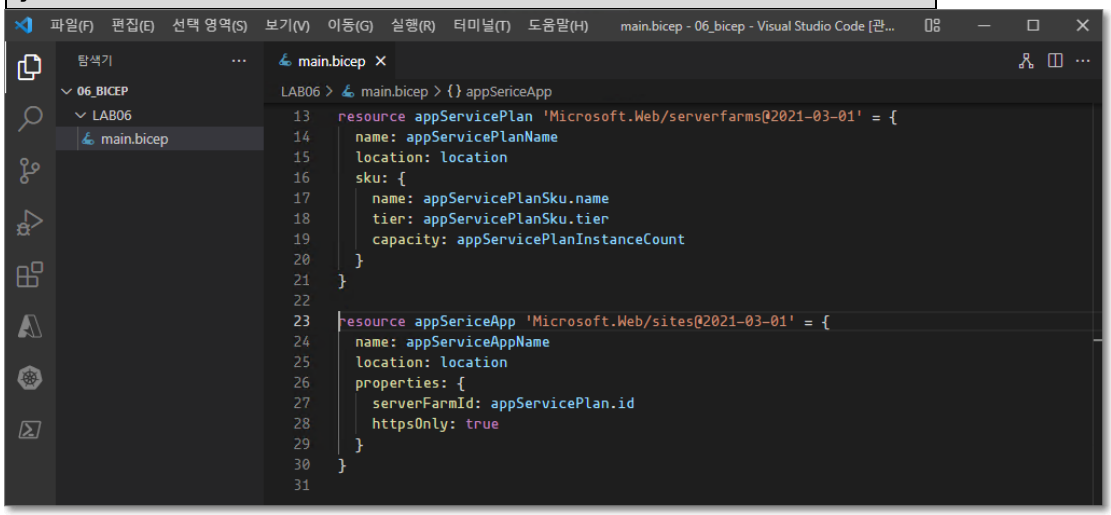
```
param environmentName string = 'dev'
param solutionName string = 'toyhr${uniqueString(resourceGroup().id)}'
param appServicePlanInstanceCount int = 1
param appServicePlanSku object = {
  name: 'F1'
  tier: 'Free'
}
param location string = resourceGroup().location

var appServicePlanName = '${environmentName}-${solutionName}-plan'
var appServiceAppName = '${environmentName}-${solutionName}-app'
```



5. "main.bicep" 파일의 제일 아래에 다음과 같이 리소스를 구성합니다.

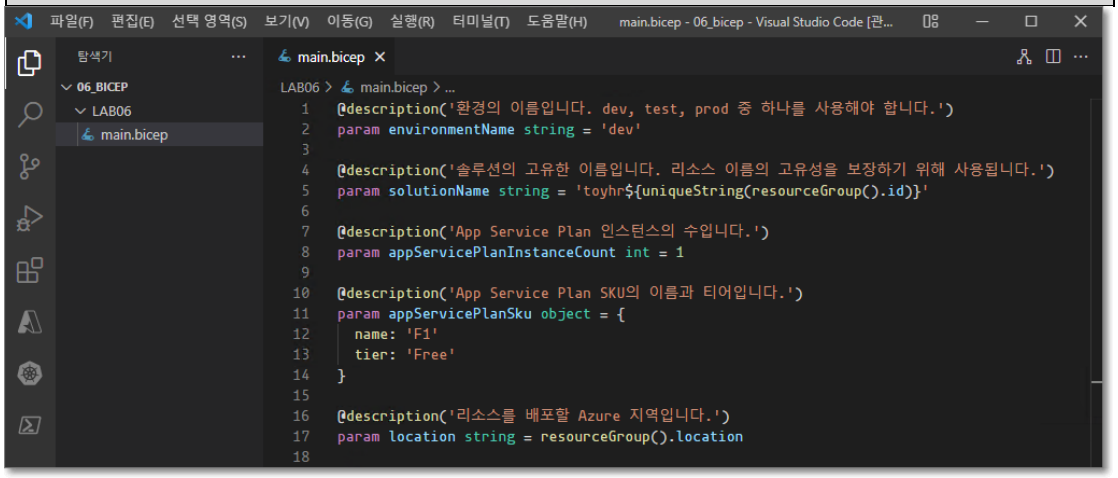
```
resource appServicePlan 'Microsoft.Web/serverfarms@2021-03-01' = {
  name: appServicePlanName
  location: location
  sku: {
    name: appServicePlanSku.name
    tier: appServicePlanSku.tier
    capacity: appServicePlanInstanceCount
  }
}

resource appSericeApp 'Microsoft.Web/sites@2021-03-01' = {
  name: appServiceAppName
  location: location
  properties: {
    serverFarmId: appServicePlan.id
    httpsOnly: true
  }
```

```
}
```



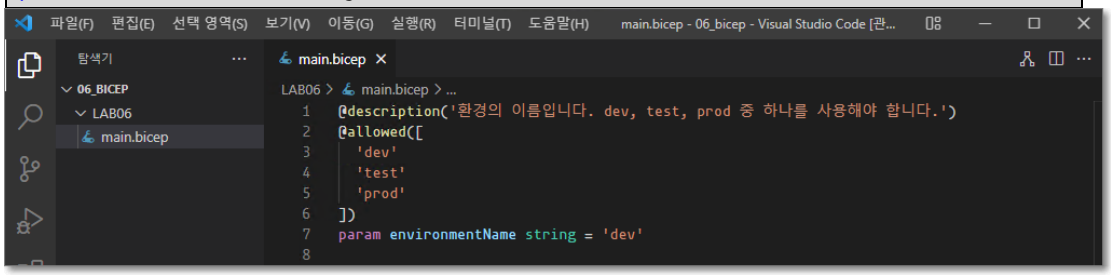
6. "main.bicep" 파일에서 앞서 만든 모든 매개 변수 바로 위에 `@description` 데코레이터를 다음과 같이 추가합니다.

```
@description('환경의 이름입니다. dev, test, prod 중 하나를 사용해야 합니다.')
param environmentName string = 'dev'

@description('솔루션의 고유한 이름입니다. 리소스 이름의 고유성을 보장하기 위해 사용됩니다.')
param solutionName string = 'toyhr${uniqueString(resourceGroup().id)}'

@description('App Service Plan 인스턴스의 수입니다.')
param appServicePlanInstanceCount int = 1

@description('App Service Plan SKU의 이름과 티어입니다.')
param appServicePlanSku object = {
  name: 'F1'
  tier: 'Free'
}

@description('리소스를 배포할 Azure 지역입니다.')
param location string = resourceGroup().location
```



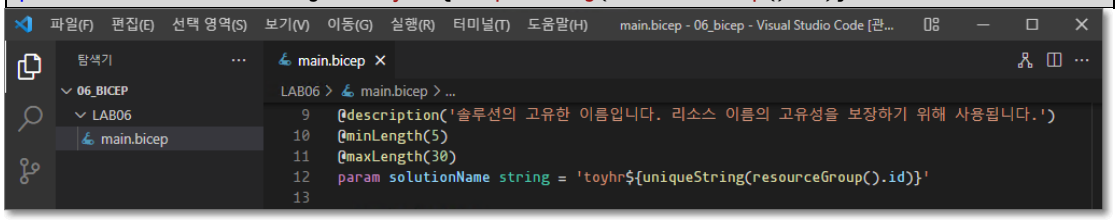
7. 앞서 선언했던 `environmentName` 매개 변수에 세가지 속성(`dev`, `test`, `prod`)만 사용할 수 있도록 제한을 설정해야 합니다. `environmentName` 매개 변수의 `@description` 데코레이터 아래에 `@allowed` 데코레이터를 삽입하고 다음과 선언합니다.
   - `environmentName` 매개 변수의 값을 `dev`, `test`, `prod`로만 제한하였습니다.
   - 향후 더 많은 환경이 추가되면 이 목록을 업데이트해야 합니다.

```
@description('환경의 이름입니다. dev, test, prod 중 하나를 사용해야 합니다.')
@allowed([
  'dev'
  'test'
  'prod'
])
param environmentName string = 'dev'
```



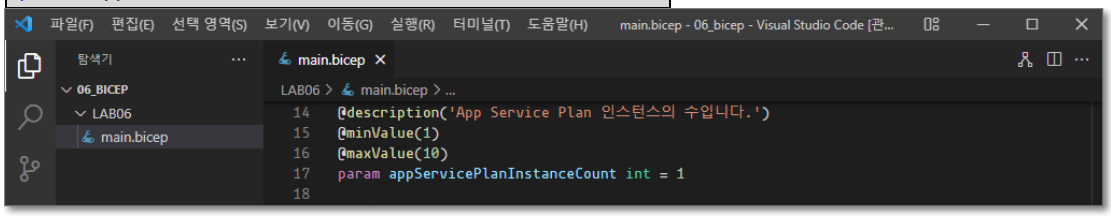
8. `solutionName` 매개 변수는 리소스의 이름을 생성하기 위해 사용됩니다. 이 매개 변수에 최소 5자에서 최대 30글자의 길이 제한을 적용하기 위해 `solutionName` 매개 변수의 `@description` 데코레이터 아래에 `@minLength`, `@maxLength` 데코레이터를 추가합니다.

```
@description('솔루션의 고유한 이름입니다. 리소스 이름의 고유성을 보장하기 위해 사용됩니다
.')
@minLength(5)
@maxLength(30)
param solutionName string = 'toyhr${uniqueString(resourceGroup().id)}'
```



9. `appServicePlanInstanceCount` 매개 변수에 1에서 10까지의 값만 허용하도록 설정하기 위해 `@description` 데코레이터 아래에 `@minValue`, `@maxValue` 데코레이터를 추가합니다.

```
@description('App Service Plan 인스턴스의 수입니다.')
@minValue(1)
@maxValue(10)
param appServicePlanInstanceCount int = 1
```



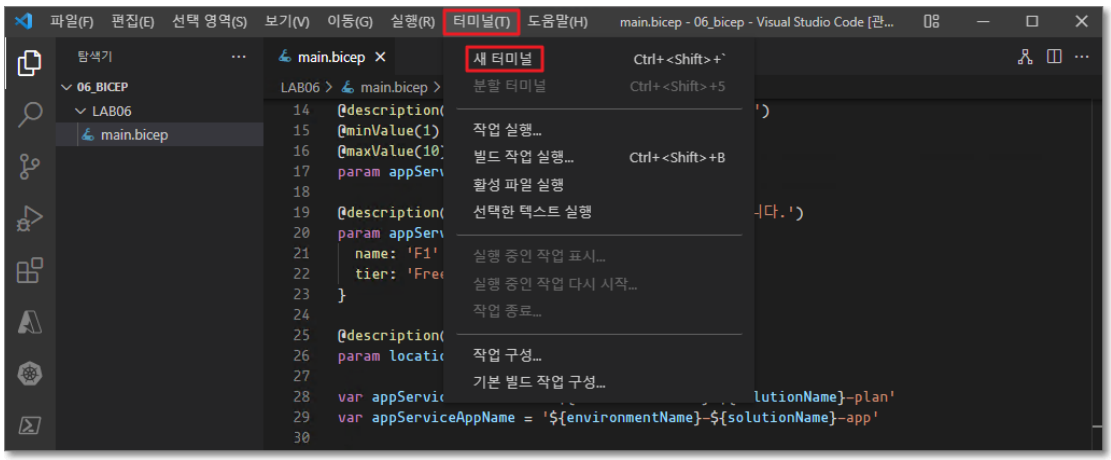
10. 작성이 완료된 Bicep 파일은 아래와 같습니다.

```
@description('환경의 이름입니다. dev, test, prod 중 하나를 사용해야 합니다.')
@allowed([
  'dev'
  'test'
  'prod'
])
param environmentName string = 'dev'

@description('솔루션의 고유한 이름입니다. 리소스 이름의 고유성을 보장하기 위해 사용됩니다
.')
@minLength(5)
@maxLength(30)
param solutionName string = 'toyhr${uniqueString(resourceGroup().id)}'

@description('App Service Plan 인스턴스의 수입니다.')
@minValue(1)
@maxValue(10)
param appServicePlanInstanceCount int = 1

@description('App Service Plan SKU 의 이름과 티어입니다.')
param appServicePlanSku object = {
  name: 'F1'
  tier: 'Free'
}

@description('리소스를 배포할 Azure 지역입니다.')
param location string = resourceGroup().location

var appServicePlanName = '${environmentName}-${solutionName}-plan'
var appServiceAppName = '${environmentName}-${solutionName}-app'

resource appServicePlan 'Microsoft.Web/serverfarms@2021-03-01' = {
  name: appServicePlanName
  location: location
  sku: {
    name: appServicePlanSku.name
    tier: appServicePlanSku.tier
    capacity: appServicePlanInstanceCount
  }
}

resource appSericeApp 'Microsoft.Web/sites@2021-03-01' = {
  name: appServiceAppName
  location: location
  properties: {
    serverFarmId: appServicePlan.id
    httpsOnly: true
  }
```

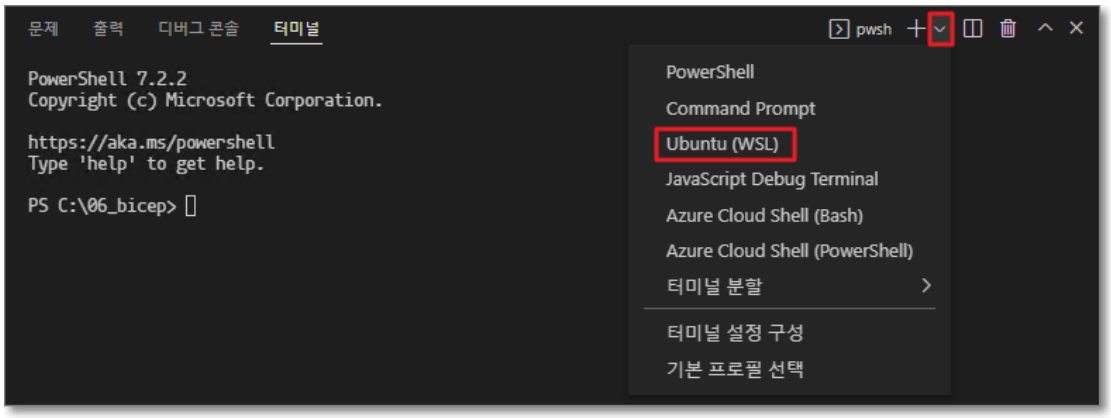
```
}
```

TASK 02. Bicep 코드 배포

1. Visual Studio Code의 메뉴에서 [터미널 - 새 터미널]을 클릭합니다.

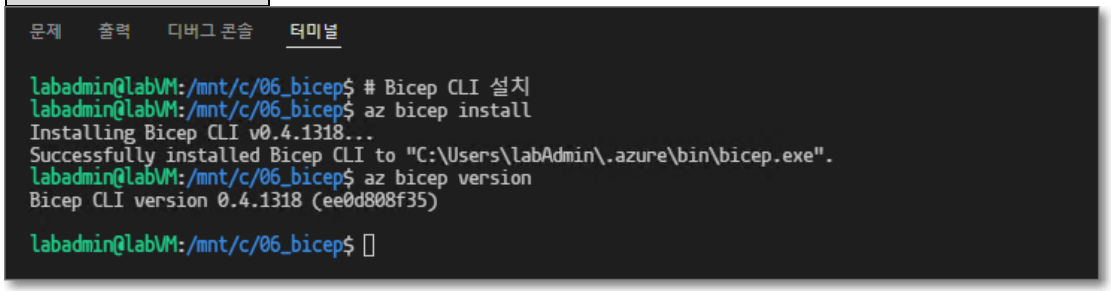


2. [터미널] 탭에서 [Launch Profile] 아이콘을 클릭한 후 [Ubuntu (WSL)]을 선택합니다.



3. [터미널]에서 다음 명령을 실행하여 Azure Bicep CLI를 설치하고 설치한 버전을 확인합니다.

```
# Bicep CLI 설치
az bicep install
az bicep version
```

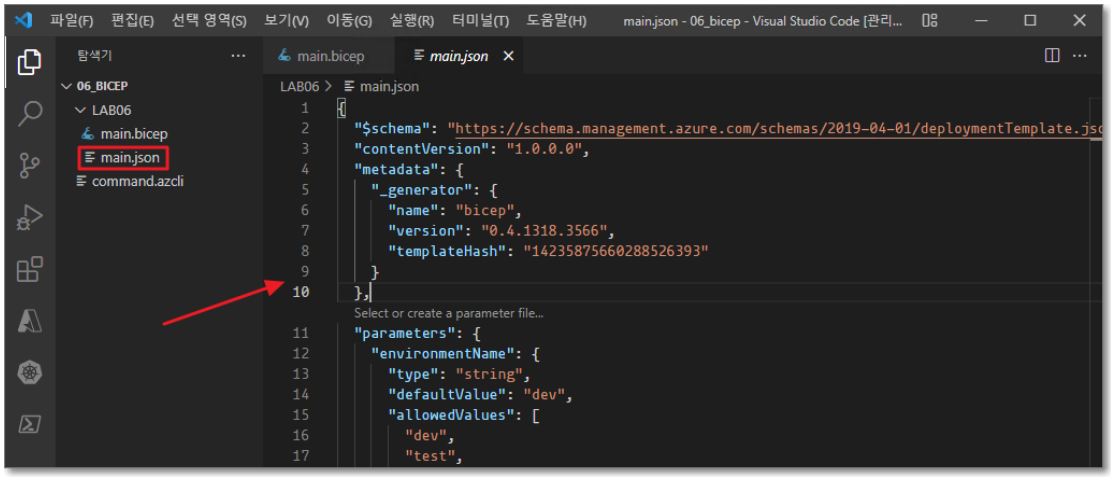


4. [터미널]에서 다음 명령을 실행하여 작성한 Bicep 파일을 ARM 템플릿 파일로 변환합니다.

```
# Bicep 컴파일
cd LAB06/
az bicep build -f main.bicep
```

```
labadmin@labVM:/mnt/c/06_bicep$ # Bicep 컴파일
labadmin@labVM:/mnt/c/06_bicep$ cd LAB06/
labadmin@labVM:/mnt/c/06_bicep/LAB06$ az bicep build -f main.bicep
labadmin@labVM:/mnt/c/06_bicep/LAB06$
```

5. Visual Studio Code의 [탐색기]에서 작성한 Bicep 파일과 동일한 이름의 json 파일이 생성된 것을 확인합니다. 이 ARM 템플릿 파일을 클릭하여 Bicep 코드와 비교해 봅니다.



6. Visual Studio Code의 터미널에서 다음 명령을 실행하여 `LAB06` 실습 폴더로 이동한 후 Azure 구독에 로그인합니다. Microsoft Edge 브라우저가 실행되면 Azure 구독에 로그인하고 브라우저를 닫습니다.

```
# Azure 구독에 로그인
az login
```

```
labadmin@labVM:/mnt/c/06_bicep/LAB06$ # Azure 구독에 로그인
labadmin@labVM:/mnt/c/06_bicep/LAB06$ az login
A web browser has been opened at https://login.microsoftonline.com/organizations/oauth2/v2.0/authorize. Please continue the login in the web browser.
 use device code flow with `az login --use-device-code`.
The following tenants don't contain accessible subscriptions. Use 'az login --allow-no-subscriptions' to have tenant level access.
[redacted] 'Contoso Lab'
[
  {
    "cloudName": "AzureCloud",
    "homeTenantId": "[redacted]",
    "id": "5515e273-1fcc-4cb2-872d-ebb448ec8979",
    "isDefault": true,
    "managedByTenants": [],
    "name": "Azure Pass - 스폰서쉽",
    "state": "Enabled",
    "tenantId": "[redacted]",
    "user": {
      "name": "labAdmin@kormtt0314.onmicrosoft.com",
      "type": "user"
    }
  }
]
labadmin@labVM:/mnt/c/06_bicep/LAB06$
```

7. 다음 명령을 실행하여 구독 목록을 확인하고 사용할 기본 구독을 선택합니다.

```
# 구독 ID 확인
az account list -o table

# 기본 구독 설정
az account set --subscription [SUBSCRIPTION_ID]
```

```
문제  출력  디버그 콘솔  터미널

labadmin@labVM:/mnt/c/06_bicep/LAB06$ # 구독 ID 확인
labadmin@labVM:/mnt/c/06_bicep/LAB06$ az account list -o table
Name                 CloudName    SubscriptionId                        State    IsDefault
-------------------  -----------  ------------------------------------  -------  -----------
Azure Pass - 스폰서쉽  AzureCloud   [redacted]                            Enabled  True
labadmin@labVM:/mnt/c/06_bicep/LAB06$
labadmin@labVM:/mnt/c/06_bicep/LAB06$ # 기본 구독 설정
labadmin@labVM:/mnt/c/06_bicep/LAB06$ az account set --subscription [redacted]
labadmin@labVM:/mnt/c/06_bicep/LAB06$
```

8. 다음 명령을 실행하여 배포에 사용할 리소스 그룹을 생성합니다.

```
# 구독에 리소스 그룹 생성
az group create -l koreacentral -n 06_bicepRg
```

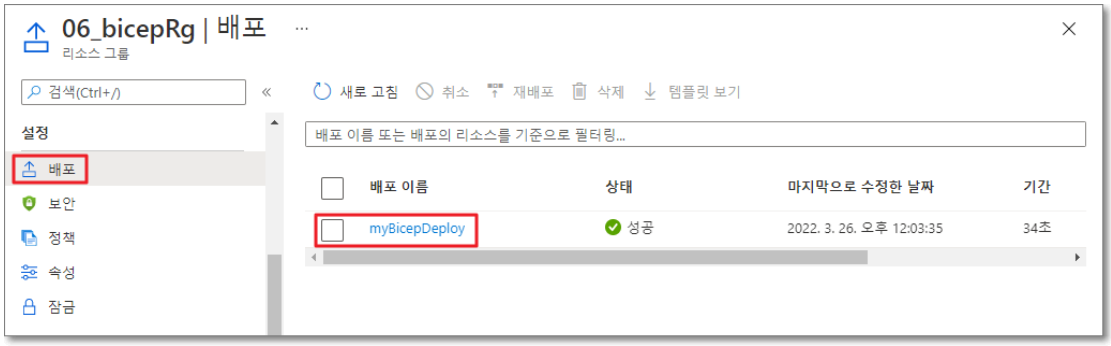
```
문제  출력  디버그 콘솔  터미널

labadmin@labVM:/mnt/c/06_bicep/LAB06$ # 구독에 리소스 그룹 생성
labadmin@labVM:/mnt/c/06_bicep/LAB06$ az group create -l eastus -n 06_bicepRg
{
  "id": "/subscriptions/[redacted]/resourceGroups/06_bicepRg",
  "location": "eastus",
  "managedBy": null,
  "name": "06_bicepRg",
  "properties": {
    "provisioningState": "Succeeded"
  },
  "tags": null,
  "type": "Microsoft.Resources/resourceGroups"
}
labadmin@labVM:/mnt/c/06_bicep/LAB06$
```

9. 다음 명령을 실행하여 Bicep 템플릿을 Azure에 배포합니다. 매개 변수에는 모두 기본값이 지정되어 있으므로 매개 변수를 따로 지정할 필요가 없습니다.

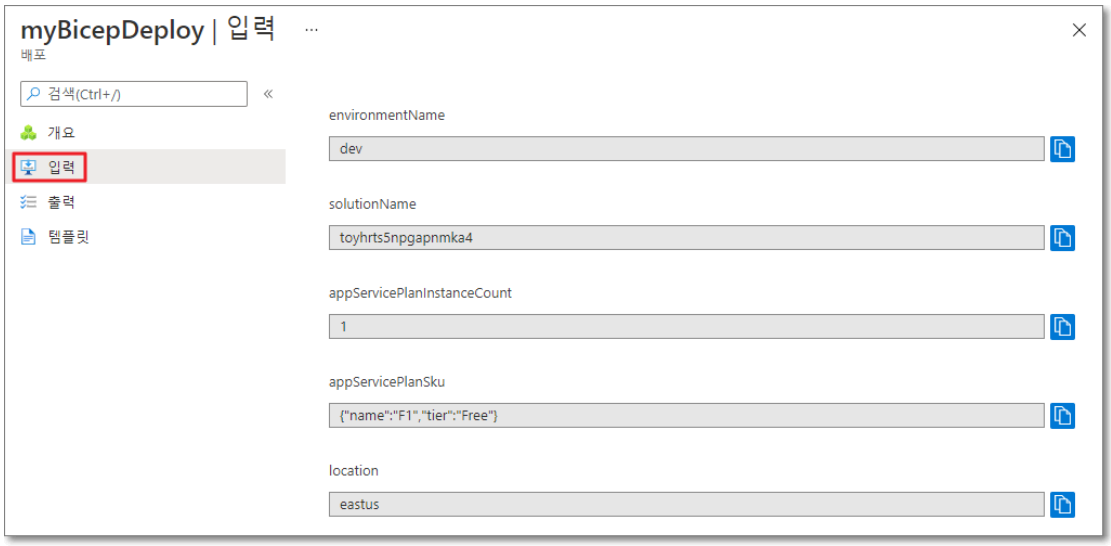
```
# Bicep 파일 배포
az deployment group create -n myBicepDeploy -g 06_bicepRg \
  -f main.bicep
```

```
문제  출력  디버그 콘솔  터미널

labadmin@labVM:/mnt/c/06_bicep/LAB06$ # Bicep 파일 배포
labadmin@labVM:/mnt/c/06_bicep/LAB06$ az deployment group create -n myBicepDeploy -g 06_bicepRg \
>   -f main.bicep
{
  "id": "/subscriptions/[redacted]/resourceGroups/06_bicepRg/providers/Microsoft.Resources/deployments/myBicepDeploy",
  "location": null,
  "name": "myBicepDeploy",
  "properties": {
    "correlationId": "35d7e919-3050-463f-8a60-5dc0e37f43e9",
    "debugSetting": null,
    "dependencies": [
      {
        "dependsOn": [
          {
            "id": "/subscriptions/[redacted]/resourceGroups/06_bicepRg/providers/Microsoft.Web/serverfarms/dev-toyhrts5
            "resourceGroup": "06_bicepRg",
            "resourceName": "dev-toyhrts5npgapnmka4-plan",
            "resourceType": "Microsoft.Web/serverfarms"
          }
        ],
```

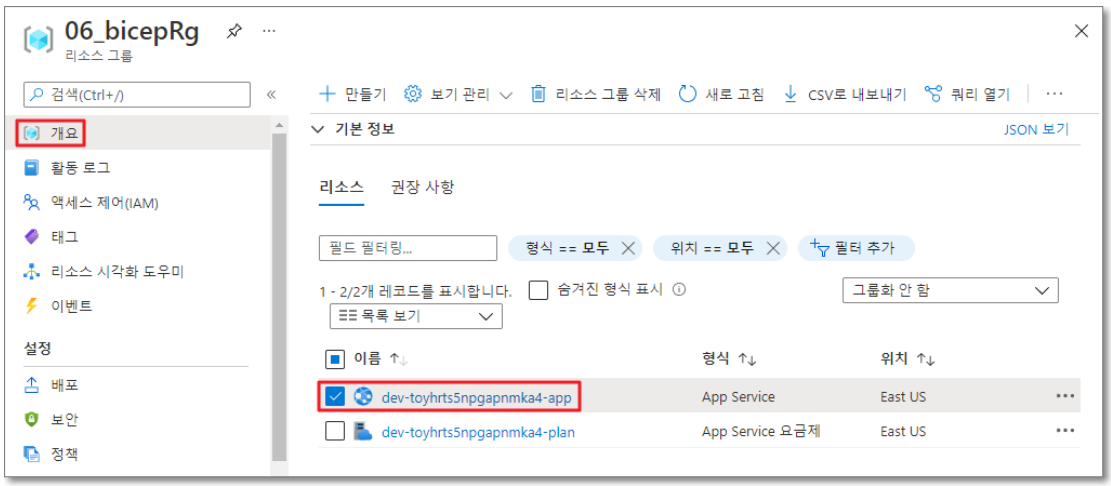
10. Azure 포털로 이동한 후 [`06_bicepRg` 리소스 그룹] 블레이드의 [설정 - 배포]로 이동합니다. `myBicepDeploy` 배포 이름을 클릭합니다.



11. [배포] 블레이드의 [입력]으로 이동한 후 아래와 같이 각 매개 변수의 값이 표시되는 것을 확인합니다.

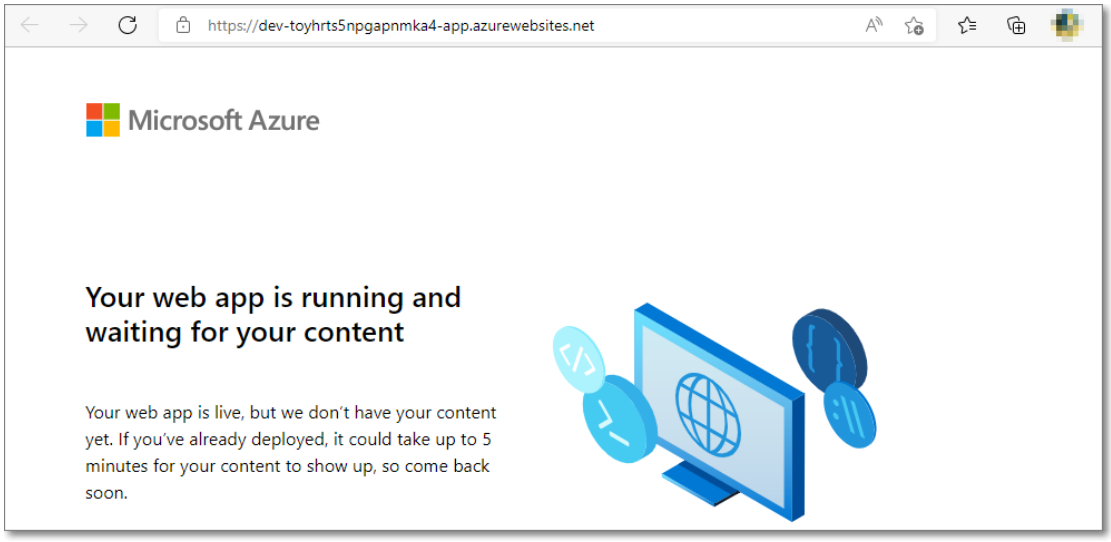


12. [`06_bicepRg` 리소스 그룹] 블레이드의 [개요]로 이동한 후 Bicep을 통해 배포한 App Service 리소스를 클릭합니다.



13. [App Service] 블레이드의 [개요]에서 URL에 표시되는 링크를 클릭합니다.

14. 아래와 같이 Bicep을 통해 App Service가 정상적으로 배포되었고 실행되는지 확인합니다.



15. Visual Studio Code로 이동한 후 [터미널]에서 다음 명령을 실행하여 리소스 그룹을 삭제합니다.

```
# 리소스 그룹 삭제
az group delete -n 06_bicepRg
```

```
문제  출력  디버그 콘솔  터미널

labadmin@labVM:/mnt/c/06_bicep/LAB06$ # 리소스 그룹 삭제
labadmin@labVM:/mnt/c/06_bicep/LAB06$ az group delete -n 06_bicepRg
Are you sure you want to perform this operation? (y/n): y
labadmin@labVM:/mnt/c/06_bicep/LAB06$
```