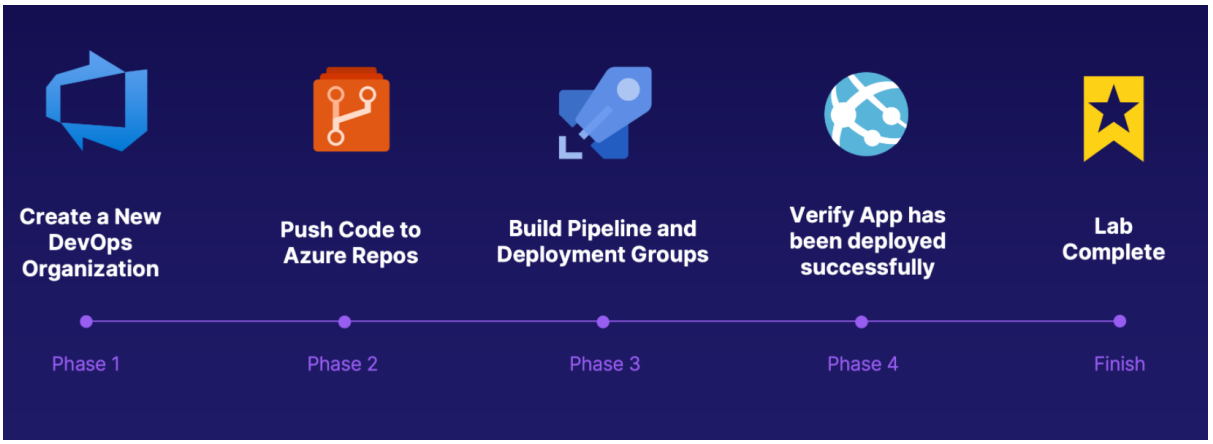
Use Deployment Groups in Azure DevOps to Deploy a .NET App

Introduction

You have a .NET application you need to deploy to a specific Azure virtual machine. You must use Azure DevOps to create a CI/CD pipeline and deploy this application using deployment groups to target that Azure VM.



Instructions

Log into the Azure portal and search for Azure DevOps from the Services menu. Create a new Azure DevOps organization and project named MyFirstProject. Select Azure Repos and import this provided code.

The application is located on the deploy branch. Import the code to deploy into your newly created Azure Repos repository. Once you've imported the code, create a build pipeline that will package up the code and ARM templates included in the repo.

Once you have your packaged application, create a release pipeline that will deploy the ARM templates, which creates a VM, a SQL server, and SQL database.

Once you've created the environment, create the deployment group.

Before deploying the web application, you will need to configure the SQL connection string using the details provided in the lab's additional resources.

Next, you can go ahead and deploy the web application to the deployment group.

Additional Resources

GitHub repository Clone URL: https://github.com/hosniah/content-az400-lab-resources
ARM Template Override parameters:

Note: Change the -databaseServerName parameter to a unique name (e.g. db1-server04042303).

```
bash -vmName "vm1" -adminUsername "cloud_user" -adminPassword "LA_ACG_Together2020!!"
-appInsightsLocation "Central US" -vmSize "Standard_D2s_v3" -location "Central US"
-windowsOSVersion "2016-Datacenter-smalldisk"  -databaseServerName "db1-server"
-databaseUsername "dbadmin" -databasePassword "LA_ACG_Together2020!!" -databaseLocation
"Central US" -databaseName "db1-db"
```

Additional SqlPackage.exe Arguments: /p:AllowIncompatiblePlatform=True

SQL Connection String details:

Note: Change the Data Source parameter to a unique name (e.g. db1-server04042303.database.windows.net).

- **Name:** defaultConnection
- **Value:** Data Source=db1-server.database.windows.net;Initial Catalog=db1-db;User ID=dbadmin;Password=LA_ACG_Together2020!!

Steps

Create an Azure DevOps Organization

1. Click the menu icon in the top left corner and select All Services.
2. Use the All services search bar to search for azure devops, then select Azure DevOps organizations.
3. On the Azure DevOps page, select My Azure DevOps Organizations.
4. On the details page, leave the default name and email address, then use the From dropdown to select your region.
5. Click Continue, then click Create new organization.
6. On the Get started with Azure DevOps window, click Continue.
7. Leave the default organization name and project host location and click Continue. Your new Azure DevOps organization is created and opens automatically.
8. After your new organization opens, create a project:
    - In the Project name field, enter *MyFirstProject*.
    - You can turn on public visibility clicking on *organization policies*. Enable 'Policies > Security Policies > *Allow public projects*' and *Save*.
    - Refresh your page and set the project's Visibility to Public.
    - Click Create project. The project is created and opens automatically.

Create a Build Pipeline

1. In the project's sidebar menu, select Repos.
2. In the Import a repository section, click Import.
3. Copy the URL provided for the lab (https://github.com/hosniah/content-az400-lab-resources) and paste it into the Clone URL field, then click Import. The code is imported from the provided GitHub repository into your project. This may take a few minutes.

4. After the code is imported, use the branch dropdown at the top of the page to select deploy.
5. (Optional) In the Files section, select the Application folder to verify the .NET application resources are there.
6. In the project's sidebar menu, select Pipelines, then click Create Pipeline.
7. Select Use the classic editor at the bottom of the page.
8. Use the default branch dropdown to select deploy, then click Continue.
9. Hover on the ASP.NET template, then click Apply.

Build the Environment

1. Select **Pipeline**. For **Agent Specification ***, use **windows-2019** to avoid warnings due to the upcoming windows-2016 environment removal.
2. Below Agent job 1 on the left, select Use NuGet 4.4.1.
3. Configure the NuGet tool installer settings on the right:
   - Change the Task Version to 1.* and the Display name to **Use NuGet 5.0.0**.
   - Expand the Advanced options and change the Version of NuGet.exe to install to **5.0.0**.
4. Select and review the settings for NuGet restore, Build solution and Test Assemblies. Leave all the default settings.
5. Add two steps after Test Assemblies:
   - After reviewing Test Assemblies, click the plus icon (+) to the right of Agent job 1.
   - On the right, use the Add tasks search bar to search for **copy**.
   - Select Copy files and click Add twice. This adds two new Copy Files to steps on the left.
6. Below Agent job 1, select the first Copy Files to step.
7. Configure the file settings on the right:
   - Change the Display name to ARM Templates.
   - Click the options menu icon to the right of the Source Folder field, then select the ArmTemplates folder and click OK.
   - Ensure the Contents field is set to ****.**
   - In the Target Folder field, enter **$(build.artifactstagingdirectory)**. This is a built-in variable which represents the artifact staging location, where artifacts are stored after the build.
8. Below Agent job 1 on the left, select the second Copy Files to step.
9. Configure the file settings on the right:
   - Change the Display name to **DACPAC**.
   - In the Contents section, enter *****.dacpac** to include all the dacpac files in your repository.
   - In the Target Folder field, enter **$(build.artifactstagingdirectory)**.
10. Below Agent job 1 on the left, select Publish symbols path, then click Remove on the right.
11. Select and review the settings for Publish Artifact. Leave all the default settings. This publishes the build so you can use it for the release.
12. Use the Save & queue dropdown to select Save & queue.
13. In the Run pipeline window, ensure **Agent Specification *** uses **windows-2019** and the deploy branch is selected. Click Save and run.

14. Select Agent job 1 to monitor the job's progress. After a few minutes, you have a successful build with 1 artifact.
15. Select the artifact link to view the artifact. You should see your application files, ARM templates, and dacpac files.

Build the Release Pipeline

Create the New Pipeline

1. Below Pipelines in the sidebar menu, select Releases, then click New pipeline.
2. In the Select a template window, select Empty job.
3. In the Artifacts section, click Add an artifact.
4. Add the artifact:
    - Ensure the Source type is set to Build and the Project is set to MyFirstProject.
    - Use the Source dropdown to select MyFirstProject-ASP.NET-CI.
    - Leave all other defaults and click Add.
5. In the Stages section, select the Stage 1 job link.

Create the ARM Template Deployment Group

1. Select **Agent job** and make sure **Agent Specification *** is using **windows-2019**.
2. Click the plus icon to the right of Agent job.
3. Use the task search bar to search for **arm**.
4. Select ARM template deployment and click Add.
5. Select the new ARM Template task on the left. Before you can complete the task details, you need to create a service connection.

Create a Service Connection

1. At the bottom of the sidebar menu, open the Project Settings in a new browser tab.
2. In the project settings sidebar, select Service connections.
3. Click Create service connection and select Azure Resource Manager.
4. Click Next, then select Service principal (manual).
5. Click Next, then create a subscription ID:
- Navigate to the Azure portal in a new tab.
- To the right of the search bar, select the Cloud Shell icon.
- In the cloud shell, select PowerShell.
- Select Show advanced settings.
- In the Storage account field, create a new storage account and enter a unique account name.
- In the File share field, enter **fileshare** as the file share name.
- Click Create storage, then expand Cloud Shell.

6. At the **PS /home/cloud>** prompt, enter **Get-AZSubscription**.
7. Below the Id section, copy the subscription ID.
8. Navigate back to the service connection details and paste the subscription ID into the Subscription Id field.

9. Navigate back to Cloud Shell and copy the subscription name, then paste it into the Subscription Name field of the service connection details.
10. In the Service Principal Id field, copy and paste the ID provided in the lab resources.
11. In the Service principal key field, copy and paste the ID provided in the lab resources.
12. Navigate back to Cloud Shell and copy the tenant ID, then paste it into the Tenant Id field of the service connection details.
13. Click Verify to verify the service connection credentials.
14. In the Service connection name, enter **SP** for service principal.
15. Click Verify and save.

Fill in the ARM Template Deployment Details

1. After the service connection is created, navigate back to your release pipeline and ensure your task is selected.
2. Click the refresh button to the right of the Azure Resource Manager connection field.
3. Use the Azure Resource Manager connection dropdown to select the SP service connection you created.
4. Use the Subscription dropdown to select your subscription.
5. Use the Resource group dropdown to select your provided resource group.
6. Check the location of the resource group:

- Navigate back to your Azure portal tab and collapse Cloud Shell.
- Click the menu icon in the top left corner, then select Resource groups.
- Your resource group details display. Note the location of the resource group.
- Navigate back to your release pipeline.

7. Use the Location dropdown to select the region you noted for your resource group.
8. Select the options menu icon to the right of the Template field.
9. Expand the **_MyFirstProject-ASP.NET-CI** (Build) and **drop** folders.
10. Select the windows-vm-sql-template.json template, then click OK.
11. In the Override template parameters field, copy and paste the parameters provided in the lab instructions. This will create a VM to host your web application, an Azure SQL server, and an Azure SQL database.
12. Change the **-databaseServerName** parameter to a unique value (e.g. **dbl-server04042303**).

Create the Azure SQL Deployment Group

1. Click the plus icon to the right of Agent job.
2. Use the task search bar on the right to search for **azure sql**.
3. Select Azure SQL Database deployment and click Add.
4. Select the new Azure SQL task on the left.
5. On the right, use the Azure Subscription dropdown to select the SP service connection you created.
6. In the Azure SQL Server field, enter **databaseServerName.database.windows.net,1433**. Be sure to replace **databaseServerName** with the unique database server name parameter you entered in the Override template parameters field. This allows your database to connect to the database server.
7. In the Database field, enter **db1-db**.

8. In the Login field, enter **dbadmin**.
9. In the Password field, copy and paste the password from the **-adminPassword** parameter provided in the lab instructions.
10. Select the options menu icon to the right of the DACPAC File field and expand all the folders until you get to the dacpac file.
11. Select the **Database1.dacpac** file and click OK.
12. In the Additional SqlPackage.exe Arguments field, enter the string **/p:AllowIncompatiblePlatform=True**.

Deploy the Release

Deploy Release 1

1. Click Save toward the top of the page, then click OK.
2. Click Create release, then click Create. A "Release-1 has been created" confirmation message displays at the top of the page.
3. Select the Release-1 link on the confirmation message and close out of the introductory window if applicable.
4. Below Stage 1, select Logs to monitor the deployment. A successful deployment takes about 7-8 minutes to complete.
5. (Optional) If you receive an error message stating that dbl-server.database.windows.net already exists, update the database server name to ensure it is unique.

- Use the Edit dropdown toward the top of the page to select Edit pipeline.
- In the Stages section, select the Stage 1 job link.
- Select Azure SQL DacpacTask and scroll down to the Azure SQL Server field.
- Update the database server name portion of the field with a unique identifier (e.g. the current date).
- Copy your new database server name and paste it into the Override template parameters code so it replaces the existing database server name.
- Click Save to save your changes, then click OK.
- Click Create release, then click Create to run the release again.
- Navigate back to the release logs to monitor the deployment and ensure it is successful.

6. After the deployment is successful, navigate back to the Azure portal and select the menu icon on the left.
7. Select All resources and review all the resources created as part of the release job.

Run the PowerShell Script

1. From the All resources list, select the vm1 virtual machine name.
2. At the top of the page, click Connect and select RDP.
3. Click Download RDP File.
4. Log in to the server through a remote desktop connection, using the credentials provided in the lab resources.
5. Navigate to the PowerShell application, then right-click and select Run as administrator.
6. Navigate back to the deployed release tab.

7. Below Pipelines in the sidebar menu, right-click on Deployment groups and open the page in a new browser tab.
8. Click Add a deployment group.
9. In the Deployment group name field, enter prod.
10. Click Create. On the right, you can see the PowerShell script you will run on your virtual machine.
11. Below the PowerShell script, check the Use a personal access token in the script for authentication checkbox.
12. Click Copy script to the clipboard.
13. Navigate back to your virtual machine and paste the PowerShell script you copied. It may take some time for the script to run.
14. When prompted, enter Y to enter deployment group tags for agent.
15. In the Enter Comma separated list of tags line, enter web.
16. When prompted to perform an unzip for tasks for each step, press Enter.
17. When prompted to enter a user account to use for the service, press Enter. The agent should then install successfully.
18. Navigate back to the Azure portal and select the Targets tab. vm1 should be in a Healthy status.

Create the Prod Deployment Group

1. Navigate back to your release pipeline.
2. Use the Edit dropdown on the right to select Edit pipeline.
3. Below Stage 1, select the tasks link.
4. Click the options menu icon to the right of Stage 1, then select Add a deployment group job.
5. Use the Deployment group dropdown to select prod.
6. In the Required tags field, enter web.

Create the Manage IISWebsite Task

1. Click the plus icon to the right of Deployment group job to add a new task.
2. Use the task search bar to search for IIS.
3. Select IIS web app manage and click Add.
4. Select the new Manage IISWebsite task on the left.
5. In the Website name field, enter Default Web Site.
6. Check the Add binding checkbox.
7. Select the menu icon to the right of the Add bindings field.
8. Review the binding details and click OK. This adds all unassigned bindings to port 80.

Create the Deploy IIS Website/App Task

1. Click the plus icon to the right of Deployment group job to add another new task.
2. Use the task search bar to search for IIS again.
3. Select IIS web app deploy and click Add.
4. Select the new Deploy IIS Website/App on the left.

5. In the Website Name field, enter Default Web Site.
6. Expand the File Transforms & Variable Substitution Options section.
7. Check the XML variable substitution checkbox. This allows you to create a variable for the SQL connection string and insert it into the web.config file.
8. Select the Variables tab along the top of the page, then click + Add.
9. In the Name field, enter defaultConnection.
10. In the Value file, copy and paste the string provided in the lab instructions. Replace the Data source parameter with the same unique database server parameter you created earlier.
11. Use the Scope dropdown to select Stage 1.
12. Select the Tasks tab along the top of the page, then select the Deploy IIS Website/App task.
13. Check the Take App Offline checkbox on the right. This allows you to take your application offline while you import the application code, then bring it back online afterwards.

Deploy Release 2

1. Click Save toward the top of the page, then click OK.
2. Click Create release, then click Create. A "Release-2 has been created" confirmation message displays at the top of the page.
3. Select the Release-2 link on the confirmation message.
4. Below Stage 1, select Logs to monitor the deployment. This may take some time to run. The deployment should be successful.
5. Navigate back to your vm1 page in the Azure portal.
6. In the sidebar menu, select Overview.
7. Copy the DNS name and open the link in a new browser tab. This may take a few minutes to load, but the website should be successfully deployed.