**Exercise - Implement the blue-green deployment pattern**

In Create a multistage pipeline by using Azure Pipelines , you built a basic deployment pipeline that deploys a web application to Azure App Service in these stages: *Dev*, *Test*, and *Staging*.

Here you add to that workflow by applying the *blue-green* deployment pattern during *Staging*.

To do so, you:

* Add a deployment slot to the App Service instance that corresponds to *Staging*.
* Add a task to the pipeline to swap the deployment slots.

**Add a deployment slot**

Here you add a deployment slot to the App Service instance that corresponds to *Staging*.

By default, every App Service instance provides a default slot, named *production*. You deployed to the *production* slot when you set up the pipeline in the previous section.

An App Service instance can have multiple slots. Here you add a second deployment slot to the App Service instance that corresponds to *Staging*. The deployment slot is named *swap*.

To add the slot:

1. Go to the Azure portal  and sign in.
2. On the menu, select **Cloud Shell**. When you're prompted, select the **Bash** experience.
3. Run the following command to get the name of the App Service instance that corresponds to *Staging* and to store the result in a Bash variable that's named staging.

**Azure CLI**

staging=$(az webapp list \

--resource-group tailspin-space-game-rg \

--query "[?contains(@.name, 'tailspin-space-game-web-staging')].{name: name}" \

--output tsv)

The --query argument uses JMESPath , which is a query language for JSON. The argument selects the App Service instance whose name field contains "tailspin-space-game-web-staging".

1. Print the staging variable to verify that you get the correct name.

**Bash**

echo $staging

Here's an example of the output:

**Output**

tailspin-space-game-web-staging-1234

1. Run the following command to add a slot named *swap* to your **staging** environment.

**Azure CLI**

az webapp deployment slot create \

--name $staging \

--resource-group tailspin-space-game-rg \

--slot swap

1. Run the following command to list your deployment slot's host name.

**Azure CLI**

az webapp deployment slot list \

--name $staging \

--resource-group tailspin-space-game-rg \

--query [].hostNames \

--output tsv

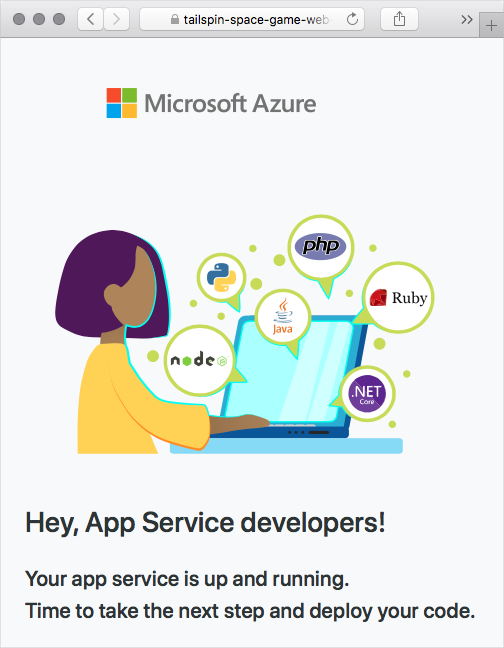
The result resembles this output:

**Output**

tailspin-space-game-web-staging-25391-swap.azurewebsites.net

**Make note of this host name for later.**

1. As an optional step, go to your site in a browser. You see the default home page because you haven't yet deployed your code to this slot.



By default, a deployment slot is accessible from the internet. In practice, you could configure an Azure virtual network that places your *swap* slot in a network that's not routable from the internet but that only your team can access. Your *production* slot would remain reachable from the internet.

**Swap deployment slots in Staging**

Here you use the AzureAppServiceManage@0  task to swap deployment slots in your *Staging* environment.

You can also use this task to start, stop, or delete a slot. Or you can use it to install site extensions or to enable continuous monitoring on App Service.

1. In Visual Studio Code, modify *azure-pipelines.yml* by using this code:

**Tip**

You can replace the entire file or just update the part that's highlighted.

**Yml**

trigger:

- '\*'

variables:

buildConfiguration: 'Release'

stages:

- stage: 'Build'

displayName: 'Build the web application'

jobs:

- job: 'Build'

displayName: 'Build job'

pool:

vmImage: 'ubuntu-18.04'

demands:

- npm

variables:

wwwrootDir: 'Tailspin.SpaceGame.Web/wwwroot'

dotnetSdkVersion: '3.1.300'

steps:

- task: UseDotNet@2

displayName: 'Use .NET Core SDK $(dotnetSdkVersion)'

inputs:

version: '$(dotnetSdkVersion)'

- task: Npm@1

displayName: 'Run npm install'

inputs:

verbose: false

- script: './node\_modules/.bin/node-sass $(wwwrootDir) --output $(wwwrootDir)'

displayName: 'Compile Sass assets'

- task: gulp@1

displayName: 'Run gulp tasks'

- script: 'echo "$(Build.DefinitionName), $(Build.BuildId), $(Build.BuildNumber)" > buildinfo.txt'

displayName: 'Write build info'

workingDirectory: $(wwwrootDir)

- task: DotNetCoreCLI@2

displayName: 'Restore project dependencies'

inputs:

command: 'restore'

projects: '\*\*/\*.csproj'

- task: DotNetCoreCLI@2

displayName: 'Build the project - $(buildConfiguration)'

inputs:

command: 'build'

arguments: '--no-restore --configuration $(buildConfiguration)'

projects: '\*\*/\*.csproj'

- task: DotNetCoreCLI@2

displayName: 'Publish the project - $(buildConfiguration)'

inputs:

command: 'publish'

projects: '\*\*/\*.csproj'

publishWebProjects: false

arguments: '--no-build --configuration $(buildConfiguration) --output $(Build.ArtifactStagingDirectory)/$(buildConfiguration)'

zipAfterPublish: true

- publish: '$(Build.ArtifactStagingDirectory)'

artifact: drop

- stage: 'Dev'

displayName: 'Deploy to the dev environment'

dependsOn: Build

jobs:

- deployment: Deploy

pool:

vmImage: 'ubuntu-18.04'

environment: dev

variables:

- group: Release

strategy:

runOnce:

deploy:

steps:

- download: current

artifact: drop

- task: AzureWebApp@1

displayName: 'Azure App Service Deploy: website'

inputs:

azureSubscription: 'Resource Manager - Tailspin - Space Game'

appName: '$(WebAppNameDev)'

package: '$(Pipeline.Workspace)/drop/$(buildConfiguration)/\*.zip'

- stage: 'Test'

displayName: 'Deploy to the test environment'

dependsOn: Dev

jobs:

- deployment: Deploy

pool:

vmImage: 'ubuntu-18.04'

environment: test

variables:

- group: 'Release'

strategy:

runOnce:

deploy:

steps:

- download: current

artifact: drop

- task: AzureWebApp@1

displayName: 'Azure App Service Deploy: website'

inputs:

azureSubscription: 'Resource Manager - Tailspin - Space Game'

appName: '$(WebAppNameTest)'

package: '$(Pipeline.Workspace)/drop/$(buildConfiguration)/\*.zip'

- stage: 'Staging'

displayName: 'Deploy to the staging environment'

dependsOn: Test

jobs:

- deployment: Deploy

pool:

vmImage: 'ubuntu-18.04'

environment: staging

variables:

- group: 'Release'

strategy:

runOnce:

deploy:

steps:

- download: current

artifact: drop

- task: AzureWebApp@1

displayName: 'Azure App Service Deploy: website'

inputs:

azureSubscription: 'Resource Manager - Tailspin - Space Game'

deployToSlotOrASE: 'true'

resourceGroupName: 'tailspin-space-game-rg'

slotName: 'swap'

appName: '$(WebAppNameStaging)'

package: '$(Pipeline.Workspace)/drop/$(buildConfiguration)/\*.zip'

- task: AzureAppServiceManage@0

displayName: 'Swap deployment slots'

inputs:

azureSubscription: 'Resource Manager - Tailspin - Space Game'

resourceGroupName: 'tailspin-space-game-rg'

webAppName: '$(WebAppNameStaging)'

sourceSlot: 'swap'

targetSlot: 'production'

action: 'Swap Slots'

Note these changes:

* + The AzureWebApp@1 task now specifies these values:
    - deployToSlotOrASE, when set to true, deploys to an existing deployment slot.
    - resourceGroupName specifies the name of the resource group. This value is required when deployToSlotOrASE is true.
    - slotName specifies the name of the deployment slot. Here you deploy to the slot that's named *swap*.
  + The new task, AzureAppServiceManage@0, swaps the deployment slots.
    - sourceSlot and targetSlot specify the slots to swap.
    - action specifies the action to take. Recall that you can use this task to start, stop, or delete a slot. Here, "Swap Slots" specifies to swap the source and target slots.

This configuration always deploys to the *swap* slot. It then swaps the *production* and *swap* slots. The swap process ensures that *production* points to the more recent deployment.

1. In the integrated terminal, add *azure-pipelines.yml* to the index. Commit the changes, and then push the branch up to GitHub.

**Tip**

Save *azure-pipelines.yml* before you run these Git commands.

**Bash**

**git add azure-pipelines.yml**

**git commit -m "Swap deployment slots"**

**git push origin blue-green**

1. In Azure Pipelines, trace the build through each of the steps.
2. As an optional step, in a browser, go to the URL that corresponds to each stage.

Although you haven't yet made changes to the website, you see that the *Space Game* website successfully deployed to each App Service environment.

