



Azure Developer Series

Migrating a dotnetcore 2-tier application to Azure,
using different architectures and DevOps best practices

Hands-On-Labs step-by-step guides

Prepared by:

Peter De Tender

CEO and Lead Technical Trainer
PDTIT and 007FFFLearning.com

@pdtit

@007FFFLearning

Version: Sept 2019 – 1.0

Information in this document, including URL and other Internet Web site references, is subject to change without notice. Unless otherwise noted, the example companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted herein are fictitious, and no association with any real company, organization, product, domain name, e-mail address, logo, person, place or event is intended or should be inferred. Complying with all applicable copyright laws is the responsibility of the user. Without limiting the rights under copyright, no part of this document may be reproduced, stored in or introduced into a retrieval system, or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), or for any purpose, without the express written permission of Microsoft Corporation.

Microsoft may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in any written license agreement from Microsoft, the furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property.

The names of manufacturers, products, or URLs are provided for informational purposes only and Microsoft makes no representations and warranties, either expressed, implied, or statutory, regarding these manufacturers or the use of the products with any Microsoft technologies. The inclusion of a manufacturer or product does not imply endorsement of Microsoft of the manufacturer or product. Links may be provided to third party sites. Such sites are not under the control of Microsoft and Microsoft is not responsible for the contents of any linked site or any link contained in a linked site, or any changes or updates to such sites. Microsoft is not responsible for webcasting or any other form of transmission received from any linked site. Microsoft is providing these links to you only as a convenience, and the inclusion of any link does not imply endorsement of Microsoft of the site or the products contained therein.

© 2019 Microsoft Corporation. All rights reserved.

Microsoft and the trademarks listed at <https://www.microsoft.com/en-us/legal/intellectualproperty/Trademarks/Usage/General.aspx> are trademarks of the Microsoft group of companies. All other trademarks are property of their respective owners.

Contents

Migrating a dotnetcore 2-tiered application to Azure using different architectures and DevOps best practices - Hands-On-Labs step-by-step	4
Abstract and Learning Objectives	5
Requirements	6
Naming Conventions:.....	6
Azure Subscription:.....	6
Other requirements:	6
Alternative Approach:	6
Final Remarks:.....	7
Lab 8: Deploying Azure Workloads using Azure DevOps	8
What you will learn	8
Time Estimate	8
Task 1: Deploying an Azure DevOps Organization.....	8
Task 2: Creating and Deploying an Azure Build Pipeline for your application.....	14
Task 3: Building a Release Pipeline in Azure DevOps	19
Task 4: Creating a Release Pipeline for Docker Containers from ACR.....	32
Task 5: Create an Azure DevOps Pipeline to deploy ACR container to Azure Kubernetes Service (AKS).....	40
Summary.....	45

Migrating a dotnetcore 2-tiered application to Azure using different architectures and DevOps best practices

- Hands-On-Labs step-by-step

You are part of an organization that is running a dotnetcore e-commerce platform application, using Windows Server infrastructure on-premises today, comprising a WebVM running Windows Server 2012 R2 with Internet Information Server (IIS) and a 2nd SQLVM running Windows Server 2012 R2 and SQL Server 2014.

The business has approved a migration of this business-critical workload to Azure, and you are nominated as the cloud solution architect for this project. No decision has been made yet on what the final architecture should or will look like. Your first task is building a Proof-of-Concept in your Azure environment, to test out the different architectures possible:

- Infrastructure as a Service (IAAS)
- Platform as a Service (PAAS)
- Containers as a Service (CaaS)

At the same time, your CIO wants to make use of this project to switch from a more traditional mode of operations, with barriers between IT sysadmin teams and Developer teams, to a DevOps way of working. Therefore, you are tasked to explore Azure DevOps and determine where CI/CD Pipelines can assist in optimizing the deployment and running operations of this e-commerce platform, especially when deploying updates to the application.

As you are new to the continuous changes in Azure, you want to make sure this process goes as smooth as possible, starting from the assessment to migration to day-to-day operations.

Abstract and Learning Objectives

This workshop enables anyone to learn, understand and build a Proof of Concept, in performing a multi-tiered .Net Core web application using Microsoft SQL Server database, platform migration to Azure public cloud, leveraging on different Azure Infrastructure as a Service, Azure Platform as a Service (PaaS) and Azure Container offerings like Azure Container Instance (ACI) and Azure Kubernetes Services (AKS).

After an introductory module on cloud app migration strategies and patterns, students get introduced to the basics of automating Azure resources deployments using Visual Studio and Azure Resource Manager (ARM) templates. Next, attendees learn about the importance of performing proper assessments, and what tools Microsoft offers to help in this migration preparation phase. Once the application has been deployed on Azure Virtual Machines, students learn about Microsoft SQL database migration to SQL Azure PaaS, as well as deploying and migrating web applications to Azure Web Apps.

After these foundational platform components, the workshop will totally focus on the core concepts and advantages of using containers for running business workloads, based on Docker, Azure Container Registry (ACR), Azure Container Instance (ACI) and WebApps for Containers, as well as how to enable container orchestration and cloud-scale using Azure Kubernetes Service (AKS).

In the last part of the workshop, students get introduced to Azure DevOps, the new Microsoft Application Lifecycle environment, helping in building a CI/CD Pipeline to publish workloads using the DevOps principals and concepts, showing the integration with the rest of the already touched on Azure services like Azure Web Apps and Azure Kubernetes Services (AKS), closing the workshop with a module on overall Azure monitoring and operations and what tools Azure has available to assist your IT teams in this challenge.

The focus of the workshop is having a Hands-On-Labs experience, by going through the following exercises and tasks:

- Deploying a 2-tier Azure Virtual Machine (Webserver and SQL database Server) using ARM-template automation with Visual Studio 2019;
- Publishing a .NET Core e-commerce application to an Azure Web Virtual Machine and SQL DB Virtual Machine;
- Performing a proper assessment of the as-is Web and SQL infrastructure using Microsoft Assessment Tools;
- Migrating a SQL 2014 database to Azure SQL PaaS (Lift & Shift);
- Migrating a .NET Core web application to Azure Web Apps (Lift & Shift);
- Containerizing a .NET Core web application using Docker, and pushing to Azure Container Registry (ACR);
- Running Azure Container Instance (ACI) and WebApp for Containers;
- Deploy and run Azure Azure Kubernetes Services (AKS);

- Deploying Azure DevOps and building a CI/CD Pipeline for the subject e-commerce application;
- Managing and Monitoring Azure Kubernetes Services (AKS);

Requirements

Naming Conventions:

IMPORTANT: Most Azure resources require unique names. Throughout these steps you will see the word “[SUFFIX]” as part of resource names. You should replace this with your initials, guaranteeing those resources get uniquely named.

Azure Subscription:

Participants need a “pay-as-you-go”, MSDN or other paid Azure subscription

- In one of the Azure Container Services tasks, you are required to create an Azure AD Service Principal, which typically requires an Azure subscription owner to log in to create this object. If you don't have the owner right in your Azure subscription, you could ask another person to execute this step for you.
- The Azure subscription must allow you to run enough cores, used by the baseline Virtual Machines, but also later on in the tasks when deploying the Azure Container Services, where ACS agent and master machines are getting set up. If you follow the instructions as written out in the lab guide, you need 12 cores.
- If you run this lab setup in your personal or corporate Azure payable subscription, using the configuration as described in the lab guide, the estimated Azure consumption costs for running the setups during the 2 days of the workshop is \$20.

Other requirements:

Participants need a local client machine, running a recent Operating System, allowing them to:

- browse to <https://portal.azure.com> from a most-recent browser;
- establish a secured Remote Desktop (RDP) session to a lab-jumpVM running Windows Server 2016;

Alternative Approach:

Where the lab scenario assumes all exercises will be performed from within the lab-jumpVM, (since several tools will be installed on the lab-jumpVM or are already installed by default), participants could also execute (most, if not all...) steps from their local client machine.

The following tools are being used throughout the lab exercises:

- Visual Studio 2017 community edition (updated to latest version); this could also be Visual Studio 2019 community edition - latest version
- Docker for Windows (updated to latest version)
- Azure CLI 2.0 (updated to latest version)
- Kubernetes CLI (updated to latest version)
- SimplCommerce Open Source e-commerce platform example (<http://www.simplcommerce.com>)

Make sure you have these tools installed prior to the workshop, if you are not using the lab-jumpVM. You should also have full administrator rights on your machine to execute certain steps within using these tools.

Final Remarks:

VERY IMPORTANT: You should be typing all of the commands as they appear in the guide, except where explicitly stated in this document. Do not try to copy and paste from Word to your command windows or other documents where you are instructed to enter information shown in this document. There can be issues with Copy and Paste from Word or PDF that result in errors, execution of instructions, or creation of file content.

IMPORTANT: Most Azure resources require unique names. Throughout these steps you will see the word "[SUFFIX]" as part of resource names. You should replace this with your initials, guaranteeing those resources get uniquely named.

Lab 8: Deploying Azure Workloads using Azure DevOps

What you will learn

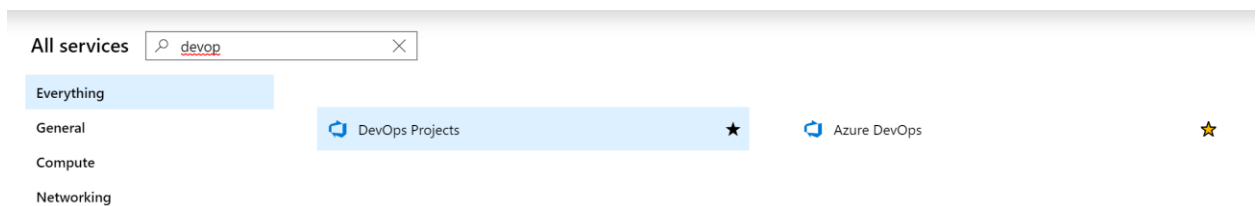
In this next lab of this workshop, you get introduced to Azure DevOps, how to deploy the service as part of your Azure subscription, as a starting point. Next, you will learn the concepts of building a CI/CD Pipeline, to deploy Azure workloads. Starting from the source code of our sample e-commerce application in a GitHub repo, you will also learn how to deploy the previously built Docker container and run this in Azure Container Instance, but deployed using Azure DevOps. Lastly, you will deploy the Docker container to the AKS Cluster you deployed earlier, again using Azure DevOps.

Time Estimate

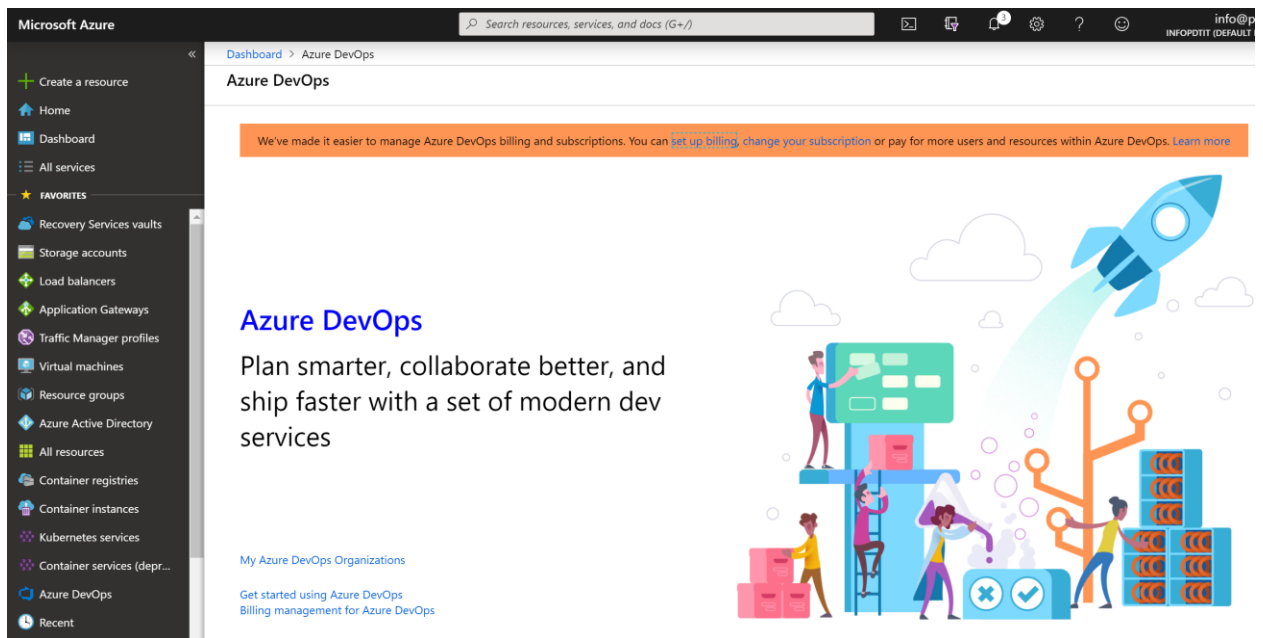
This lab shouldn't take longer than 60 minutes.

Task 1: Deploying an Azure DevOps Organization

1. From your Azure Portal, **browse to all services**, and search for **DevOps**.

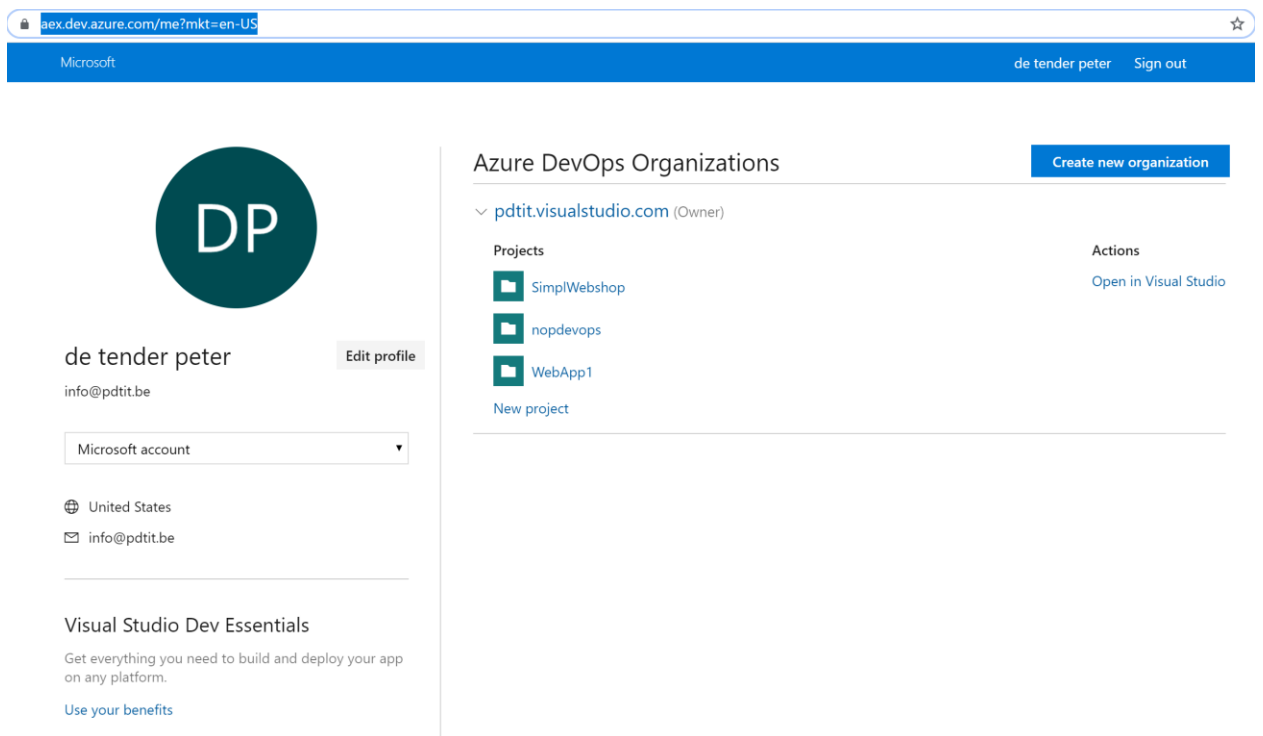


2. Select "Azure DevOps"

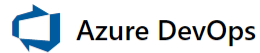


3. Next, click “My Azure DevOps Organizations”

After providing your Azure credentials once more, you will get redirected to a visualstudio.com website, and eventually ending up at <https://aex.dev.azure.com/me?mkt=en-US>



- Here, select "Create New Organization"
-



info@pdtit.be

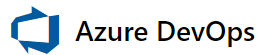
Get started with Azure DevOps

Choosing **Continue** means that you agree to our [Terms of Service](#), [Privacy Statement](#), and [Code of Conduct](#).

- ☐ I would like information, tips, and offers about Azure DevOps and other Microsoft products and services. [Privacy Statement](#).

Continue

-
- Press Continue



Almost done...

Name your Azure DevOps organization

dev.azure.com/ pdtitws123

We'll host your projects in

West Europe



Continue

6. Press Continue, and provide a name for your DevOps Project

Create a project to get started

Project name *

devopsdemo



Visibility



Public

Anyone on the internet can view the project. Certain features like TFVC are not supported.




Private

Only people you give access to will be able to view this project.





+ Create project


7. Confirm by clicking the "+ Create Project" button; your Azure DevOps "Workspace" gets created


 **Azure DevOps**


pdtitws123 / devopsdemo / Overview / Summary


 **devopsdemo** +


 Overview


 **Summary**


 Dashboards


 Wiki


 Boards

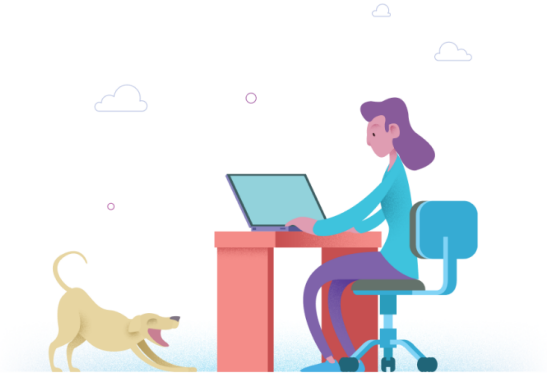
 Repos

 Pipelines

 Test Plans

 Artifacts

 **devopsdemo**



Welcome to the project!

What service would you like to start with?

Boards

Repos

Pipelines

Test Plans

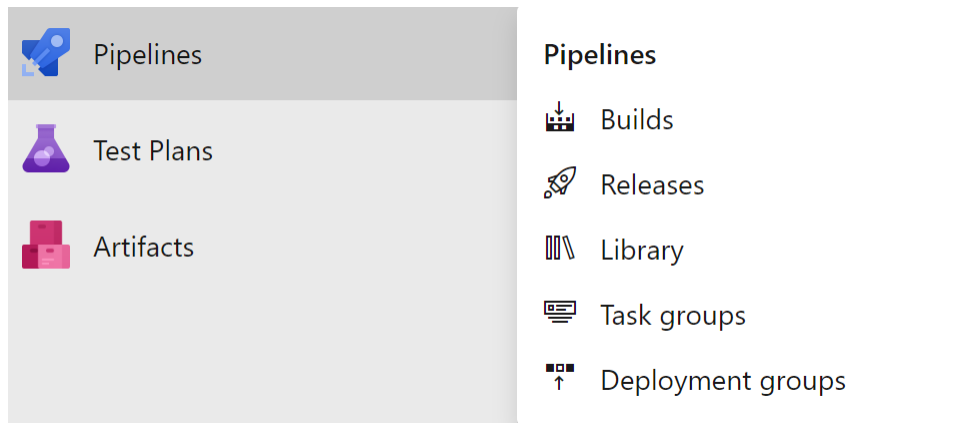
Artifacts

8. This completes the task, in which you deployed Azure DevOps and configured an Azure DevOps Organization. In the next task, you will build your DevOps Pipelines.

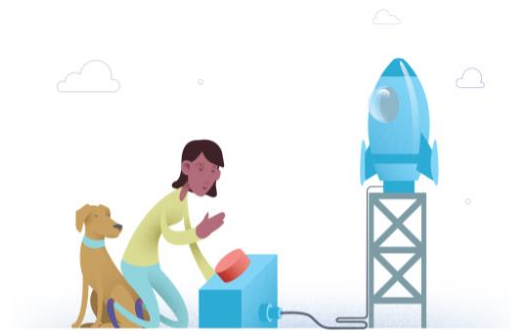
Task 2: Creating and Deploying an Azure Build Pipeline for your application

While Azure DevOps gives you an end-to-end solution to manage your application development and deployment lifecycle, this lab focuses mainly on the **Azure Pipelines** service within.

1. Select "Pipelines"



2. Select "Builds"



No build pipelines were found

Automate your build in a few easy steps with a new pipeline.

[New pipeline](#)

3. Click "New Pipeline"

Where is your code?



Azure Repos Git YAML

Free private Git repositories, pull requests, and code search



Bitbucket Cloud YAML

Hosted by Atlassian



GitHub YAML

Home to the world's largest community of developers



GitHub Enterprise Server YAML

The self-hosted version of GitHub Enterprise



Other Git

Any generic Git repository



Subversion

Centralized version control by Apache

4. You are prompted for selecting your version control repository. Select GitHub

✓ Connect

Select

Configure

Review

New pipeline

Select a repository

Filter by keywords

My repositories



007FFFLearning/SimplDev

Aug 13



007FFFLearning/Nopcommercedevops

Aug 10



pdtit/TechdaysFinland

Mar 3



pdtit/TDFinland

private

Mar 3

Note: you will be prompted to create an integration between Azure DevOps and Github, by authentication with your GitHub account. If you don't have one yet, go to <http://www.github.com>, and create one.

Once you have your GitHub account ready, you can create a new GitHub repo, and upload the SimplCommerce source files into it. Another option is cloning the source files from the sample repo I am using here 007FFFLearning/SimplDev

check the GitHub website for instructions on how to clone an existing repo into your own GitHub.

5. In the next step, you are presented with an Azure DevOps Pipeline.yml file; as you know, this file is automatically being build up, based on the detection of the source code language.

✓ Connect

✓ Select

✓ Configure

Review

New pipeline

Review your pipeline YAML

azure-pipelines.yml

```
1  # ASP.NET Core
2  # Build and test ASP.NET Core projects targeting .NET Core.
3  # Add steps that run tests, create a NuGet package, deploy, and more:
4  # https://docs.microsoft.com/azure/devops/pipelines/languages/dotnet-core
5
6  trigger:
7    - master
8
9  jobs:
10   - job: Linux
11     pool:
12       vmImage: 'Ubuntu 16.04'
13     steps:
14       - task: DotNetCoreInstaller@0
15         inputs:
16           packageType: 'sdk'
17           version: '2.2.401'
18       - script: dotnet build ./SimplCommerce.sln
19         displayName: 'dotnet build'
20       - script: ./run-tests.sh
21         displayName: 'run tests'
22       - task: PublishTestResults@2
23         displayName: 'Publish Test Results **/*.trx'
24         condition: succeededOrFailed()
25         inputs:
26           testResultsFormat: VSTest
27           testResultsFiles: '**/*.trx'
```

6. For example here, it recognizes the source code as **.NET Code**, and will create several build pipelines, for different platforms (Linux, MacOS, Windows). If you want, you could remove parts of this pipeline file, and only testing against Linux OS for example.
7. Confirm by clicking **Run**; your pipeline build process is starting

Logs Summary Tests

Linux
Not started

macOS
Running

Windows
Not started

LinuxRelease
Not started

macOS

Pool: Azure Pipelines · Agent: Hosted Agent

Started: 9/5/2019, 5:07:19 PM

8s

Initialize job · succeeded	1s
Checkout · succeeded	5s
DotNetCoreInstaller	2s

```

Starting: DotNetCoreInstaller
=====
Task       : .NET Core SDK/runtime installer
Description: Acquire a specific version of the .NET Core SDK from the internet or local cache and add it to the PATH
Version    : 0.2.4
Author     : Microsoft Corporation
Help       : https://docs.microsoft.com/azure/devops/pipelines/tasks/tool/dotnet-core-tool-installer
=====

```

8. Wait for this process to complete successfully.

#20190813.1: initial version
 Release
All logs

Manually run aug 13 at 11:49 pm by de tender peter 007FFFLearning/SimplDev master e7801ad

Logs Summary Tests

Linux
Succeeded

macOS
Succeeded

Windows
Succeeded

LinuxRelease
Succeeded

Linux

Pool: Hosted Ubuntu 1604 · Agent: Hosted Agent

Started: 8/13/2019, 11:50:04 PM

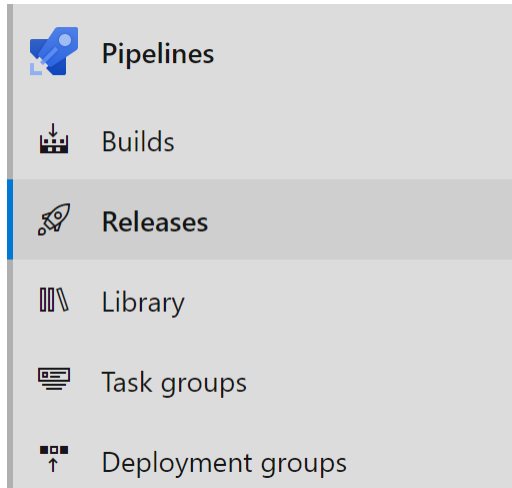
2m 45s

Prepare job · succeeded	<1s
Initialize job · succeeded	2s
Checkout · succeeded	6s
DotNetCoreInstaller · succeeded	8s
dotnet build · succeeded	1m 52s
run tests · succeeded	33s
Publish Test Results **/*.trx · succeeded	1s
Post-job: Checkout · succeeded	<1s
Finalize Job · succeeded	<1s

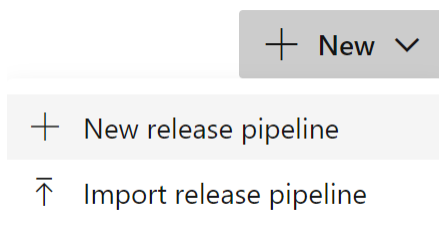
9. This completes the task in which you set up a Build Pipeline, based on application source code in GitHub. In the next task, we will continue the process, by creating and running a release pipeline, publishing the code to Azure.

Task 3: Building a Release Pipeline in Azure DevOps

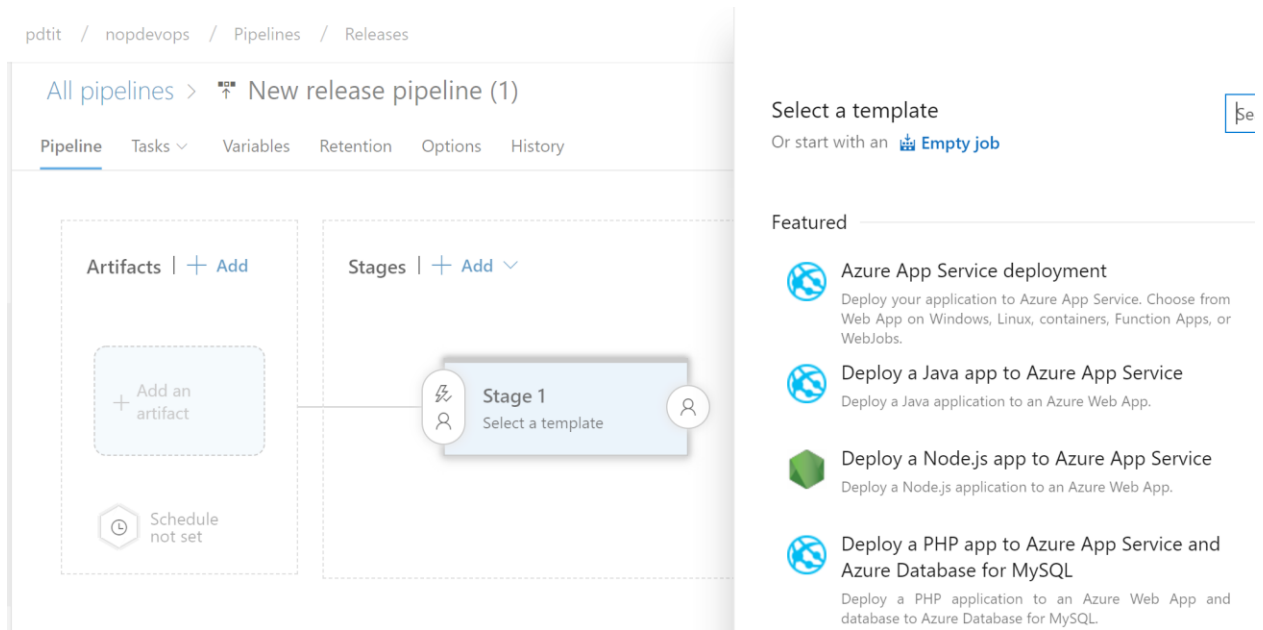
1. From Azure DevOps, Pipelines, select **Releases**



2. Next, select **New / Release Pipeline**



3. This launches the New release pipeline creating wizard.



4. From the **Template list**, select **Azure App Service Deployment**

Provide a description for the **Stage Name**, for example `Deploy_to_webapp`

×

Stage

Deploy_to_webapp

🗑️ Delete

📁 Move

⋮

📄 Properties

^

Name and owners of the stage

Stage name

Deploy_to_webapp

Stage owner

DP

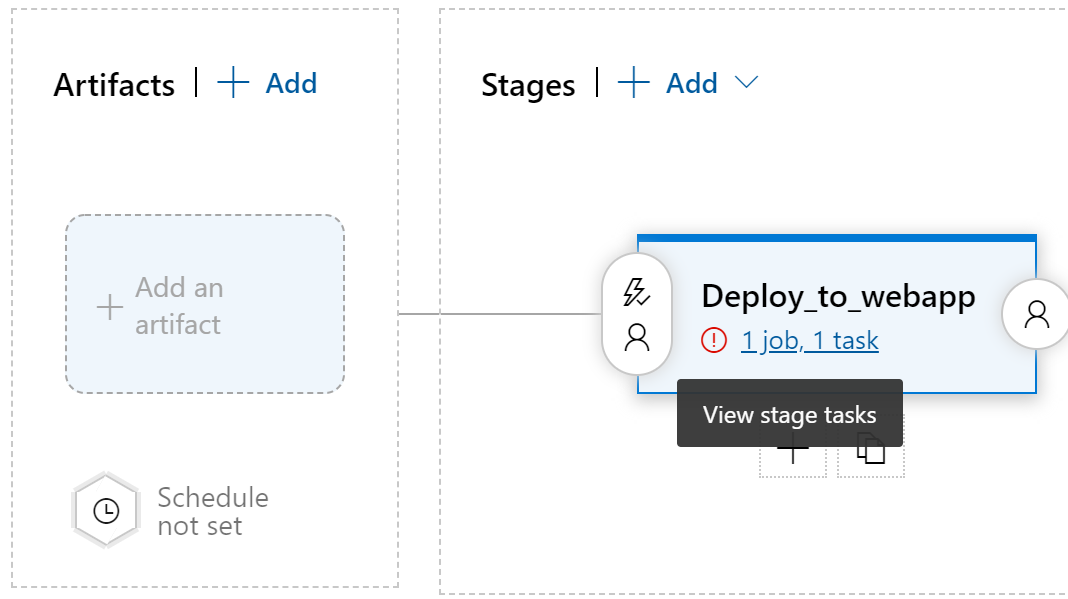
de tender peter

×

5. The Pipeline now looks like this:

All pipelines > New release pipeline (1)

Pipeline  Tasks  Variables Retention Options History



6. In the **Stage** field, select “1 job, 1 task”; this is where you will provide the settings of the Azure Web App environment you will use for the actual deployment.

Stage name

Deploy_to_webapp

Parameters ⓘ | [Unlink all](#)

Azure subscription * 🔗 | [Manage](#) 🔗



ⓘ This setting is required.

App type 🔗

Web App on Windows



App service name * 🔗



ⓘ This setting is required.

7. Complete the settings according to your Azure subscription and resources you already have from earlier deployments.

Stage name

Deploy_to_webapp

Parameters ⓘ | [Unlink all](#)

Azure subscription * [Manage](#)

007FFFLearning Labs (0a407898-c077-442d-8e17-71420aa82426) ▼



ⓘ Scoped to subscription '007FFFLearning Labs'

App type [Manage](#)

Web App on Windows ▼

App service name * [Manage](#)

SimplWebAppv1 ▼



8. When done, click **Pipeline** in the top menu of your Azure Pipeline project, to return and complete the next step in the Pipeline process, specifying the Artifacts.

[All pipelines](#) > **New release pipeline (1)**

Pipeline Tasks ▾ Variables Retention Options History

Artifacts | [+ Add](#)

+ Add an artifact

Schedule not set

Stages | [+ Add](#) ▾



Deploy_to_webapp

1 job, 1 task

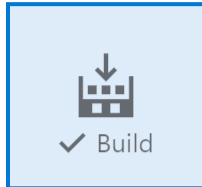


9. Select Artifacts / Add an Artifact

10. Complete the parameters, according to the Build Pipeline we created in the previous task:

Add an artifact

Source type



Azure Repos ...



GitHub



TFVC

5 more artifact types 

Project * 

nopdevops



Source (build pipeline) * 


007FFFLearning.SimplDev



Default version * 

Latest



Source alias * 

_007FFFLearning.SimplDev



No version is available for **007FFFLearning.SimplDev** or the latest version has no artifacts to publish. Please check the source pipeline.

Add

11. Confirm by clicking "Add"

12. When done, click the "Save" button

13. Next, click the "Create Release" button

All pipelines > New release pipeline (1)

Save Create release

Pipeline Tasks Variables Retention Options History

Artifacts | [+ Add](#)

_007FFFLearning.S
implDev

Schedule
not set

Stages | [+ Add](#)

Deploy_to_webapp
1 job, 1 task

14. Save the changes, and confirm with "Create"


Create a new release






New release pipeline (1)

Pipeline ^

Click on a stage to change its trigger from automated to manual.


 Deploy_to_we

Stages for a trigger change from automated to manual. 

 Deploy_to_webapp 

Artifacts ^


Select the version for the artifact sources for this release

Source alias	Version	
_007FFFlearning.SimplDev	20190813.1	

Release description

15. You get prompted the release is getting created

[All pipelines](#) >  New release pipeline (1)

 Release **Release-1** has been created

Pipeline


Tasks ▾


Variables


Retention


Options



History

Artifacts |  Add


_007FFFLearning.S
implDev

 Schedule
not set

Stages |  Add ▾


 Deploy_to_webapp
1 job, 1 task


16. Click "Release-1", which redirects you to the details of the Release process

↑ New release pipeline (1) > Release-1 ▾

Pipeline Variables History | + Deploy ▾ ⓧ Cancel ↺ Refresh ✎ Edit ▾ ...


Release

Manually triggered


by  de tender peter

9/5/2019, 5:34 PM

Artifacts

_007FFFLearning.Simpl...

20190813.1

 master

Stages

Deploy_to_webapp

☐ Not deployed


17. Select "Deploy_to_webapp", and confirm "Deploy"; the status will change to "in progress"

↑ New release pipeline (1) > Release-1 ▾

Pipeline Variables History | + Deploy ▾ ⏹ Cancel ↺ Refresh ✎ Edit ▾ ...


Release

Manually triggered

by  de tender peter


9/5/2019, 5:34 PM

Artifacts



[_007FFFLearning.Simpl...](#)

[20190813.1](#)

 master

Stages

Deploy_to_webapp

In progress

1/1 tasks

Initialize job

⌚ 00:08

18. Wait for this process to initialize

↑ New release pipeline (1) > Release-1 > Deploy_to_webapp ▾ In progress

← Pipeline Tasks Variables Logs Tests | ☁ Deploy ⏹ Cancel ↺ Refresh ⬇ Download all logs ✎ Edit ▾ ...

Deployment process

In progress

Run on agent

In progress

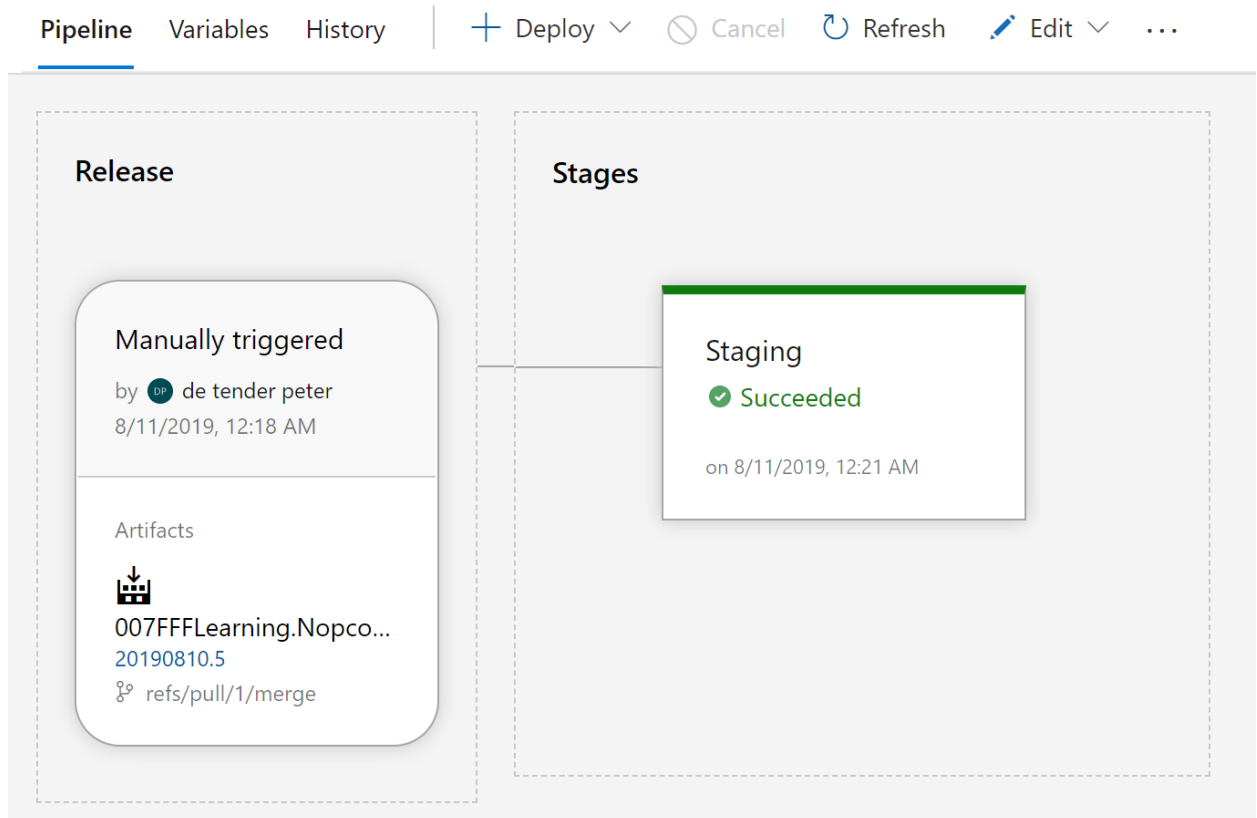
Run on agent

Pool: Azure Pipelines · Agent: Hosted Agent

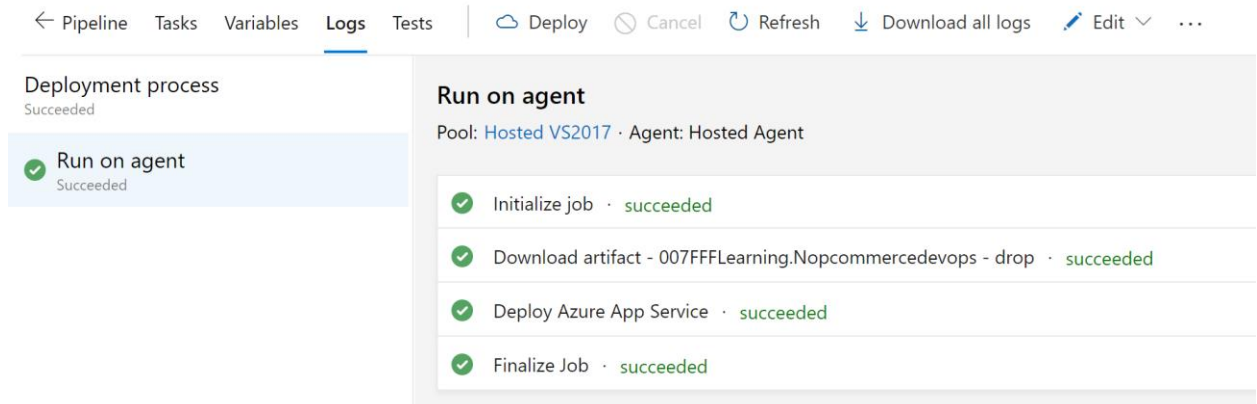
Initialize job

Waiting for console output from an agent...

19. Wait for the release stage to complete successfully



20. Selecting the **Staging**, will open the details of the release process



21. From the Azure Portal, browse to the **Azure WebApp** you selected in the **Release Pipeline** as target, and validate it is running as expected.



New products

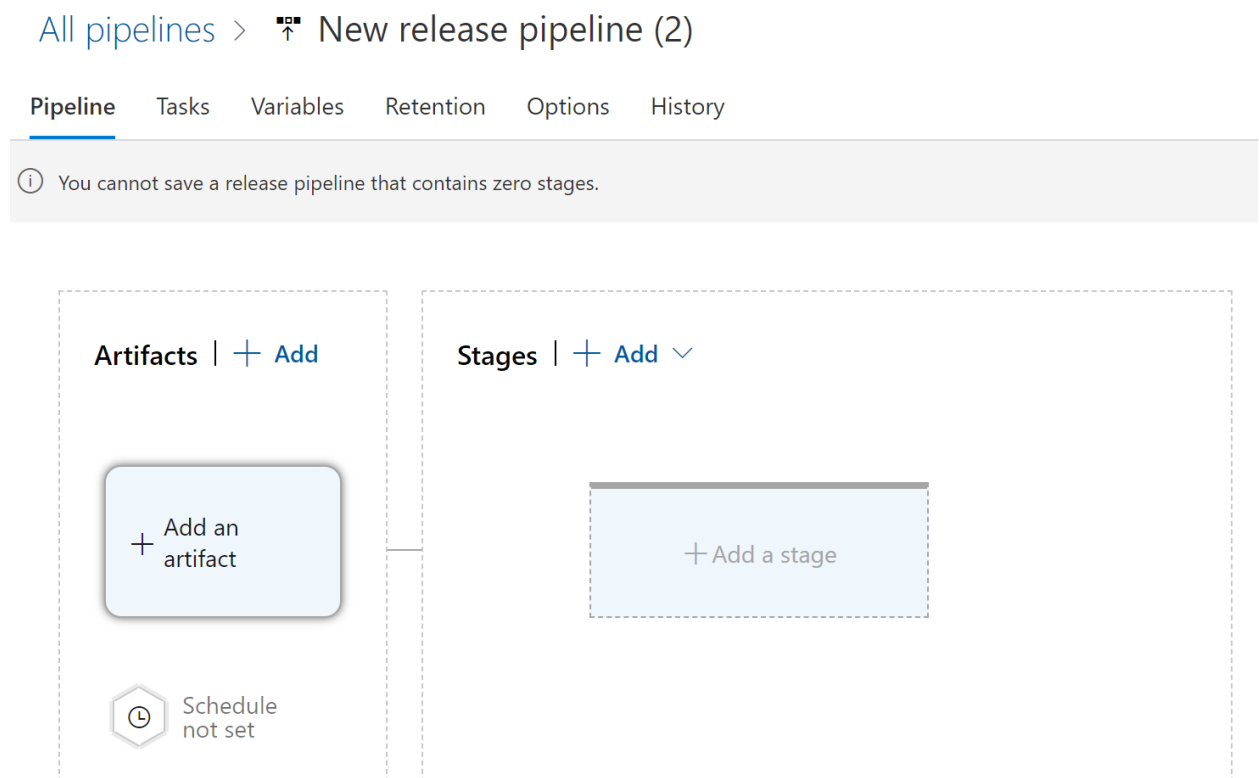


22. This completes the task in which you created a Release Pipeline, based on a previous Build Pipeline configuration.

Task 4: Creating a Release Pipeline for Docker Containers from ACR

Similar to the previous Release Pipeline from source code in GitHub to a published Azure WebApp, we can use the same concept to create a release pipeline, based on a Docker container in Azure Container Registry. This is similar to the manual task you ran in lab 4 earlier.

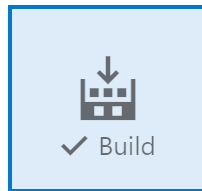
1. From **Azure DevOps**, select **Pipeline / Release / New Release Pipeline**
2. When the Template window appears, close it, and select **Artifacts** first



3. Choose "Add an Artifact"; From the Artifact blade, click "5 more artifact types", to extend the list of artifacts to choose from.

Add an artifact

Source type



5 more artifact types ▾

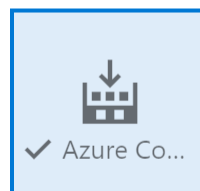
Project * ⓘ

nopdevops

4. Select **"Azure Container Registry"**, and complete the parameters according to the existing resources in your Azure subscription, reusing the resources from previous lab exercises (Azure Container Registry, Repository,...)

Add an artifact

Source type



Jenkins

Service connection * | [Manage](#)

007FFFLearning Labs (0a407898-c077-442d-8e17-71420aa82426)



Resource Group * ⓘ

noprg



Azure Container Registry * ⓘ

nopacr1



Repository * ⓘ

simplpdtv1



Default version * ⓘ


Latest

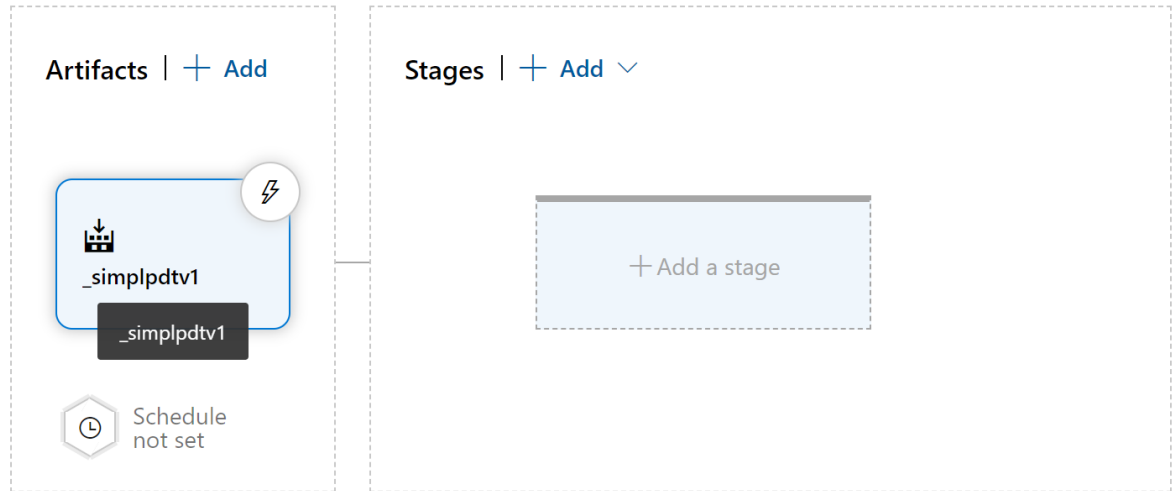


Source alias * ⓘ

_simplpdtv1

5. Confirm the artifacts selection, by clicking **Add**
6. Your artifact will be completed

 You cannot save a release pipeline that contains zero stages.



- Next, click Stage / Add a Stage, and select Azure App Service deployment

Select a template

Or start with an  Empty job

 cont

Others



Azure App Service deployment with continuous monitoring

Deploy your Web applications to Azure App Service and enable continuous monitoring using Application Insights.

Featured



Azure App Service deployment

Deploy your application to Azure App Service. Choose from Web App on Windows, Linux, containers, Function Apps, or WebJobs.

Apply

- Confirm with Apply

9. Provide the required parameters, based on the resources in your Azure environment

Azure Web App for Containers ⓘ

 View YAML  Remove

Task version 1.* ▼

Display name *

Azure Web App on Container Deploy: contnopwebapp

Azure subscription * ⓘ | [Manage](#) ↗

007FFFLearning Labs (0a407898-c077-442d-8e17-71420aa82426) ▼



ⓘ Scoped to subscription '007FFFLearning Labs'

App name * ⓘ

contnopwebapp ▼



☐ Deploy to Slot or App Service Environment ⓘ

Image name ⓘ

nopacr1.azurecr.io/nopcommerce_420_source_nopcommerce_web:\$(build.buildnumber)

Configuration File ⓘ



Startup command ⓘ

Application and Configuration Settings ▼

Control Options ▼

10. Note that you have to manually provide the correct string for both Azure Container Registry and Repository you will use

App service name *

A globally unique top-level domain name for your specific registry or namespace.
Note: Fully qualified image name will be of the format: '**<registry or namespace>/<repository>:<tag>**'. For example, '**myregistry.azurecr.io/nginx:latest**'.

This field is linked to 1 setting(s) across:
'Run on agent': 'Deploy Azure App Service'

Repository *

This setting is required.

11. Press "Save", followed by "Create Release"
12. This kicks off the release creation; follow the different steps occurring, and wait for them to complete successfully

```

✓ Azure Web App on Container Deploy: contnopwebapp
1  [2019-08-11T18:16:40.7798636Z] ##[section]Starting: Azure Web App on Container Deploy: contnopwebapp
2  2019-08-11T18:16:41.0218887Z =====
3  2019-08-11T18:16:41.0219066Z Task           : Azure Web App for Containers
4  2019-08-11T18:16:41.0219314Z Description    : Deploy containers to Azure App Service
5  2019-08-11T18:16:41.0219368Z Version       : 1.0.20
6  2019-08-11T18:16:41.0219430Z Author        : Microsoft Corporation
7  2019-08-11T18:16:41.0219492Z Help          : https://docs.microsoft.com/azure/devops/pipelines/tasks/deploy/azure-rm-web-app-containers
8  2019-08-11T18:16:41.0219597Z =====
9  2019-08-11T18:16:42.4716693Z Got service connection details for Azure App Service: 'contnopwebapp'
10 2019-08-11T18:16:42.7336601Z Single-container Deployment to the webapp 'contnopwebapp' as only the image detail was specified.
11 2019-08-11T18:16:43.1923367Z Updating App Service Configuration settings. Data: {"appCommandLine":null,"linuxFxVersion":"DOCKER|nopacr1.azurecr.io/nopcon
12 2019-08-11T18:16:45.4837837Z Updated App Service Configuration settings.
13 2019-08-11T18:16:45.4843895Z Restarting App Service: contnopwebapp
14 2019-08-11T18:16:46.2856398Z App Service 'contnopwebapp' restarted successfully.
15 2019-08-11T18:16:51.5119795Z Successfully updated App Service configuration details
16 2019-08-11T18:16:53.0316534Z Successfully updated deployment History at https://contnopwebapp.scm.azurewebsites.net/api/deployments/71565547411856
17 2019-08-11T18:16:53.6187573Z App Service Application URL: http://contnopwebapp.azurewebsites.net
18 2019-08-11T18:16:53.6565928Z ##[section]Finishing: Azure Web App on Container Deploy: contnopwebapp
19

```


13. Once the task is complete, you can see its overall status from the Pipeline window

Release_from_ACR > Release-2

Pipeline Variables History | + Deploy Cancel Refresh Edit ...


Release

Manually triggered

by  de tender peter

8/11/2019, 9:16 PM

Artifacts

 _nopcommerce_420_so...

latest

Stages

Deploy Container

✓ Succeeded

on 8/11/2019, 9:16 PM

14. Check back in Azure WebApps if your app is running successfully, by connecting to the Azure WebApp URL for this Azure resource

simplcontwebapp.azurewebsites.net

SimplCommerce

Search here...

All C

WOMAN MAN ▼ SHOES WATCHES



New products



simplcontwebapp.azurewebsites.net/man



15. This completes the task in which you created a new Azure Pipeline Release, deploying an Azure Web App for Containers, relying on a repository in Azure Container Registry.

Task 5: Create an Azure DevOps Pipeline to deploy ACR container to Azure Kubernetes Service (AKS)

In this scenario, you will create yet another Azure Release Pipeline, this time pushing a container from ACR into the earlier deployed Azure Kubernetes Service cluster.

1. From **Azure DevOps**, select **Pipeline / Release / New Release Pipeline**
2. Close the appearing **template window**, and return to the **Artifacts**
3. Repeat the steps from the previous task, selecting **Azure Container Registry** as source, and selecting the **ACR registry and container repository you want to use** for this deployment..

Service connection * | [Manage](#)

007FFFLearning Labs (0a407898-c077-442d-8e17-71420aa82426)

Resource Group * ⓘ

noprg

Azure Container Registry * ⓘ

nopacr1

Repository * ⓘ

simplpdtv1

Default version * ⓘ

Latest

Source alias * ⓘ

_simplpdtv1

Add

- Next, select “Add Stage”, and select the Deploy to a Kubernetes Cluster” template

Select a template

Or start with an  Empty job

 kub



Featured



Deploy to a Kubernetes cluster

Deploy, configure, update your containerized applications to a Kubernetes cluster.


Apply

Others

- Confirm by clicking Apply

[All pipelines](#) >  New release pipeline (3)

Pipeline

Tasks 

Variables

Retention


Options

History

Stage 2


Deployment process

Agent job

 Run on agent





kubectl

 Deploy to Kubernetes

- Select “kubectl”, and provide the necessary settings and parameters of the AKS cluster you deployed in a previous lab, knowing you **only need to provide the Kubernetes Service Connection** name

Deploy to Kubernetes ⓘ

 [View YAML](#)  [Remove](#)

 Task version 

Display name *

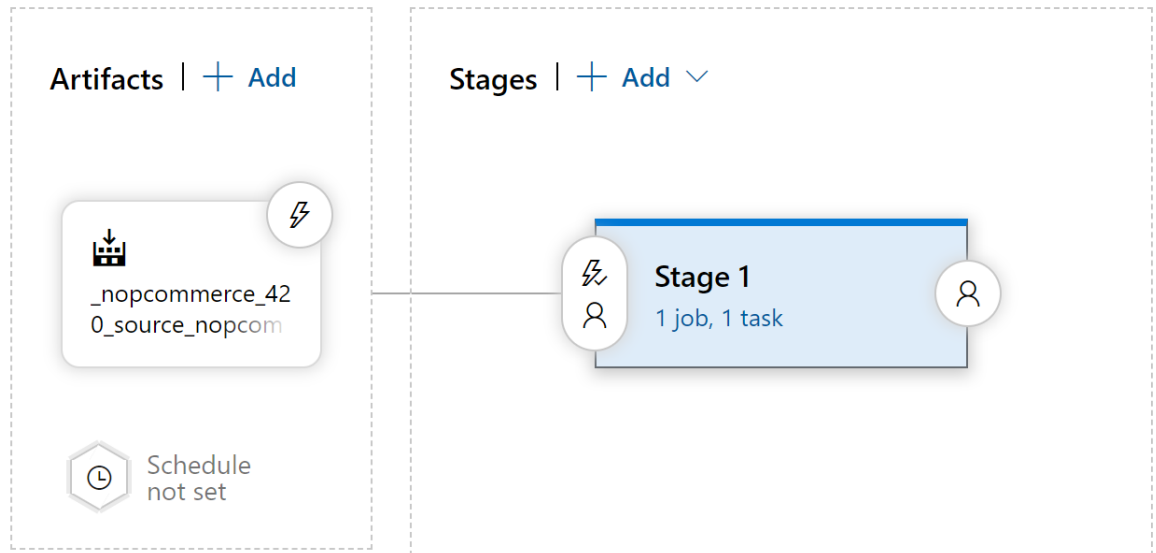
Kubernetes service connection ⓘ | [Manage](#) 

  [+ New](#)

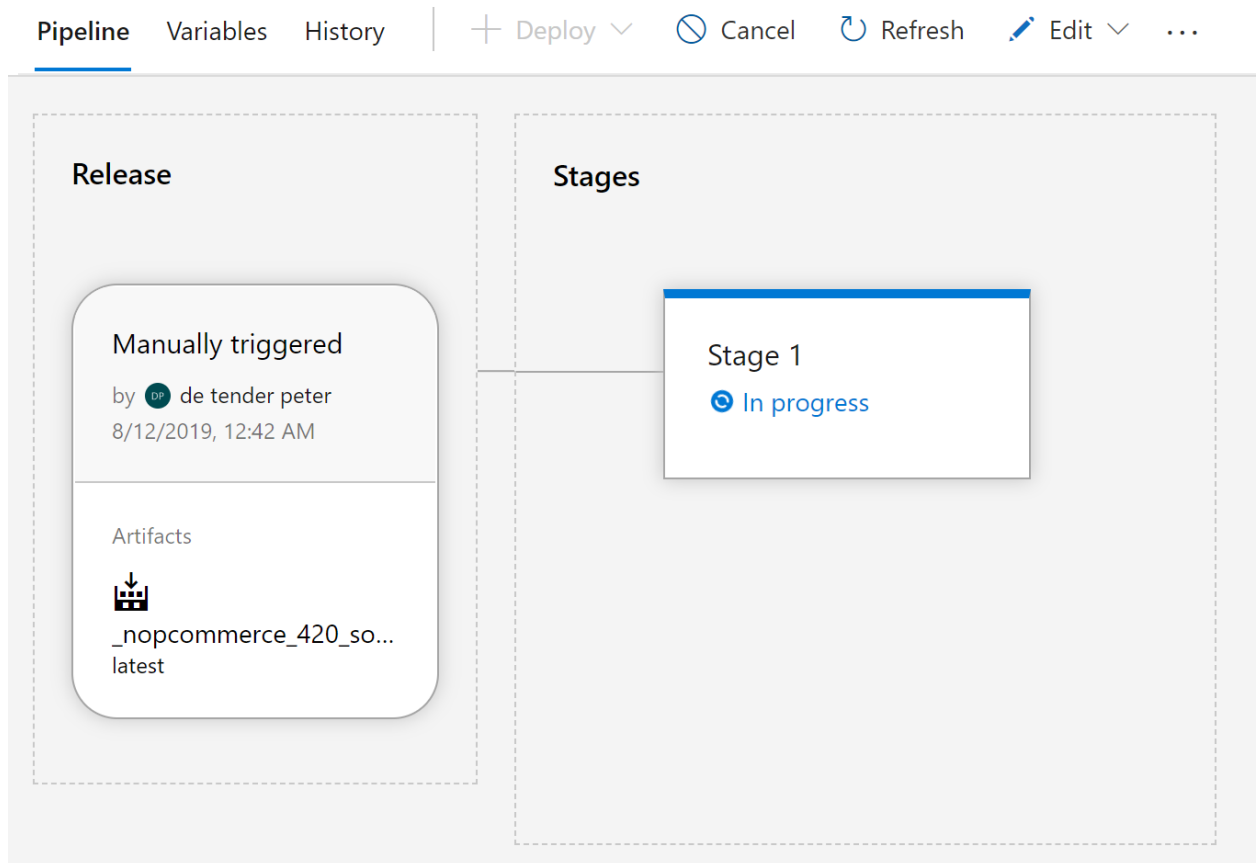
Namespace ⓘ

NOTE: The full deployment process is much much much more powerful and providing many more settings that we will do here, but it is mainly to allow you experiencing what a base deployment Release Pipeline can do, and how to configure it.

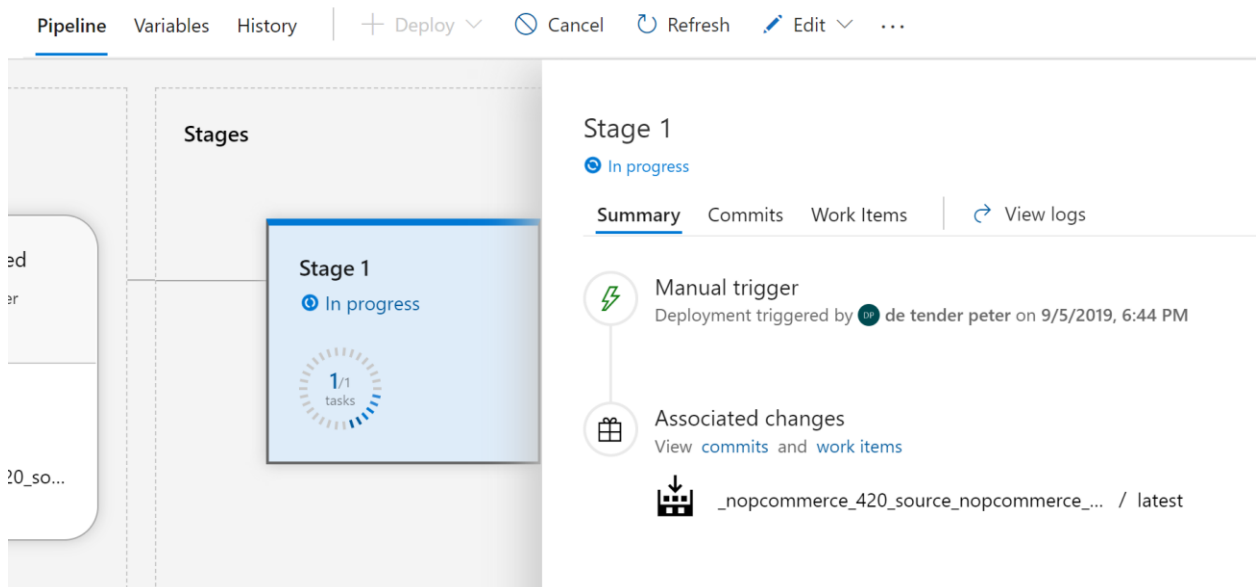
7. **Save** the settings, and validate the Pipeline



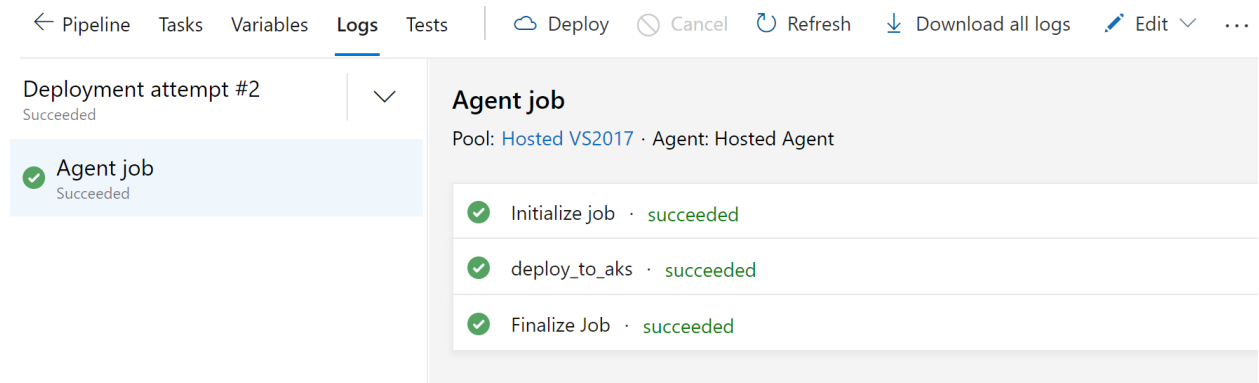
8. Run the Release



9. Seelct "In Progress"



10. Wait for the task to complete successfully



11. This completes the task in which you created a new Azure Release Pipeline for a deployment of an ACR-stored repository, to an existing Azure Kubernetes Cluster.

Summary

In this lab, you performed several tasks around Azure DevOps, starting from the initial creation of an Azure DevOps Organization, followed by creating an Azure DevOps Build Pipeline, using a GitHub repository with an application's source code. In the next task, you created a Release Pipeline, deploying the Build from the previous task, publishing an Azure Web App.

The following tasks involved creating an Azure DevOps Release Pipeline to publish an Azure Container Registry repository image to Azure Container Instance, as well as publishing to Azure Kubernetes Service.

Congrats if you completed all labs with all tasks from all modules. You should now have a real good understanding of Azure and where it can help in your overall digital transformation. Reach out when having any questions, concerns, or want to share overall feedback about the workshop content. Have a nice day!